Maia-2: A Unified Model for Human-AI Alignment in Chess

Zhenwei Tang
University of Toronto
josephtang@cs.toronto.edu

Difan Jiao
University of Toronto
difanjiao@cs.toronto.edu

Reid McIlroy-Young
Harvard University
reidmcy@seas.harvard.edu

Jon Kleinberg
Cornell University
kleinberg@cornell.edu

Siddhartha Sen
Microsoft Research
sidsen@microsoft.com

Ashton Anderson
University of Toronto
ashton@cs.toronto.edu

Abstract

There are an increasing number of domains in which artificial intelligence (AI) systems both surpass human ability and accurately model human behavior. This introduces the possibility of algorithmically-informed teaching in these domains through more relatable AI partners and deeper insights into human decision-making. Critical to achieving this goal, however, is coherently modeling human behavior at various skill levels. Chess is an ideal model system for conducting research into this kind of human-AI alignment, with its rich history as a pivotal testbed for AI research, mature superhuman AI systems like AlphaZero, and precise measurements of skill via chess rating systems. Previous work in modeling human decision-making in chess uses completely independent models to capture human style at different skill levels, meaning they lack coherence in their ability to adapt to the full spectrum of human improvement and are ultimately limited in their effectiveness as AI partners and teaching tools. In this work, we propose a unified modeling approach for human-AI alignment in chess that coherently captures human style across different skill levels and directly captures how people improve. Recognizing the complex, non-linear nature of human learning, we introduce a *skill-aware attention mechanism* to dynamically integrate players' strengths with encoded chess positions, enabling our model to be sensitive to evolving player skill. Our experimental results demonstrate that this unified framework significantly enhances the alignment between AI and human players across a diverse range of expertise levels, paving the way for deeper insights into human decision-making and AI-guided teaching tools. Our implementation is available here.

1 Introduction

There are an increasing number of domains in which artificial intelligence (AI) systems both surpass human ability and accurately model human behavior. This combination of machine mastery over a domain and computational understanding of human behavior in it introduces the possibility of algorithmically-informed teaching and learning. AI-powered aids could guide people along reliable and efficient improvement paths, synthesized from their knowledge of both human trajectories and

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

objective performance. Relatable AI partners, on the other hand, could learn to act alongside human counterparts in synergistic and complementary ways.

Researchers have begun to tackle this challenge in the model system of chess. Once held to be an ideal testbed for developing artificial intelligence, it is now the perfect domain to pursue human-AI alignment. The AI community finally surpassed all human ability in chess approximately 20 years ago, a milestone achievement and watershed cultural moment. Now, superhuman AI chess engines are ubiquitous and widely used. Despite this transformation, chess has never been more popular, becoming a mainstream activity in many countries during the last few years. There is now both unprecedented demand for chess education, as well as mature superhuman AI that could in principle help meet it.

However, existing models fall short of being effective learning tools and relatable partners. Traditional chess engines such as Stockfish and AlphaZero are unimaginably strong, but they don't play in ways that humans can easily understand or learn from. Comparing one's own decisions with those of traditional engines, it is easy to see how near-perfect AI would have improved upon your play but hard to see how you could realistically do the same. Recent work has resulted in the development of Maia, a suite of models that aim to mimic human behavior in chess at various skill levels by learning to predict actual human moves from a wealth of online gameplay data [1]. While substantially more human-like, these models still cannot power effective algorithmic teaching tools because of several limitations.

First and foremost, Maia models players at different skill levels completely independently; games by players at one skill level and those by an adjacent skill level are fed into separate instances of the same architecture and result in separate models. This has the downside that predictions from one model are independently made of predictions from any other. Viewed as a whole, they are volatile: the Maia models might predict that at one level players will approach a position correctly, then at the next level they will make a horrible mistake, then at the next level they will do fine again, and so on. In a word, they fail to *cohere*. People don't improve along volatile paths, they steadily get better. The unrealistically incoherent predictions made by separate models don't suggest realistic pathways that people can take in order to get better. In order to serve as algorithmic teachers or learning aids, our models of human behavior must be coherent.

Building a coherent model of human skill in chess is difficult, because the breadth of skill in chess is almost incomprehensibly large. Decisions made by beginners bear only the faintest of relations to those made by masters. A difference of 200 points in chess rating systems roughly equates to a 75% win rate for the higher-rated player—typically higher than the best record of any team in the entire National Basketball Association. On the online chess platform we study, there are players who are 2600 rating points apart—or 13 successive steps of 75%-vs.-25% dominance apart from each other. Capturing this breadth of skill in a single model, in a coherent, smooth fashion, is a challenge.

We contribute a unified modeling approach for human-AI alignment in chess that coherently captures human style across different skill levels and directly captures how people improve. Since our model builds directly on original Maia, we call it *Maia-2*. Maia-2 consists of a standard residual network tower that processes chess positions into features, and our novel contribution of a *skill-aware attention module* with channel-wise patching. This innovation takes the position representation outputted by the residual network tower and simple player skill encodings and learns how player skill levels interact with chess positions to produce the moves humans make. Unlike previous models, Maia-2 only requires the current board position as input (as opposed to six), which dramatically reduces training time and increases flexibility (e.g. for applying the model in non-game contexts where there may be no 6-board history). In addition to policy and value heads like in previous work, we also add an additional auxiliary information head that helps the model learn a deeper understanding of human chess moves.

We evaluate Maia-2 along two key dimensions: move prediction *accuracy* and *coherence*. Testing it against the original Maia models, Stockfish, and AlphaZero, Maia-2 emerges as the most accurate human move predictor by far, surpassing original Maia by almost 2 full percentage points. Analyzing move prediction accuracy by skill level, Maia-2 matches and surpasses all other models on all skill levels. Furthermore, Maia-2's gains in perplexity are similarly striking, reducing average perplexity from a previous record of 4.67 bits down to 4.07 bits. Maia-2 achieves these accuracy gains while being substantially more coherent than the original Maia models. For example, call a model's treatment of a position *monotonic* if it assigns a monotonically increasing probability to the

correct move as we increase skill. While original Maia treats 1% of a random sample of positions monotonically, Maia-2 treats a remarkable 27% of the same positions monotonically. This is in keeping with our intuitive understanding of how chess players steadily and smoothly improve across the skill range. Finally, we conduct an investigation of the human chess concepts Maia-2 learns and varies with skill via linear probes, and find that skill-dependent concepts like overall board evaluation indeed vary with skill, but skill-independent concepts do not, which also accords with our understanding of how human players make decisions.

2 Related Work

Chess and AI. This paper draws on the long history of chess at the forefront of AI research [2, 3, 4, 5]. We engage with 3 distinct approaches to building chess AI: heuristic [6], learned [7], and textual [8]. *Heuristic search:* The original approach to computer chess was heuristics-based [4, 9]. This method was famously used by IBM's Deep Blue to defeat Garry Kasparov [2] and is currently used by Stockfish [6], one of the strongest chess engines in the world. *Learned search:* Alpha(Zero) Go [7, 10] is a set of neural networks that learn to play Go with methods that generalized to other games, including chess, with AlphaZero [10]. Chess AI with learned search is also extended to multi-agent systems [11], where diverse AI systems can outperform a single AI in challenging tasks such as chess. *Chess as text:* Large language models [12, 13, 14] have recently been found to perform well on tasks that the models were not *explicitly* trained on [15], including playing chess without fine-tuning [16, 17, 18]. This has lead to chess knowledge being one of the tested features in BIG-Bench [19], a popular LLM evaluation suite. Additionally, fine-tuning a language model can lead to systems that not only play chess, but can also generate comments, describe positions, and create other simple analyses of a game [20, 21, 8].

Human-AI Alignment in Chess. Building a chess engine that can defeat any human has been a solved problem for over 20 years. This has led to a new research agenda in extracting useful knowledge from these superhuman systems. A direct way of doing this is to probe an AI chess engine in a human representation space. Without any prior human knowledge or guidance, evidence of human chess concepts learned by AlphaZero is found and measured by linear probes [22]. Going further, AlphaZero also encodes knowledge that extends beyond existing human knowledge but is ultimately learnable by humans [23].Another direction was the creation of a 'behavioral stylometry' model that can identify chess players from the moves they play [24]. An alternative approach to creating systems that can act as guides to humans is demonstrated by Maia [1, 25], in which a model is trained to predict the next move a human will play, instead of optimizing for winning the game. In addition to predicting human actions the models have been fine-tuned to predict a given player's actions [25]. The prediction accuracy can be improved via a reinforcement learning-style search [26].

3 Methodology

We propose a unified model architecture to capture human decision-making in chess across a broad spectrum of skill levels. Since this model builds upon the previous Maia move-matching models, we call it *Maia-2*. As shown in Figure 1, Maia-2 first encodes active and opponent skill levels and the chess positions, respectively. Then the encoded skill levels and positions are fused using our skill-aware attention with channel-wise patching architecture. The fused representations are then used for move prediction (policy head), auxiliary information prediction (auxiliary head), and game outcome prediction (value head). We now discuss each of these components in detail.

3.1 Skill Level Encoder

Instead of directly incorporating player ratings as numerical inputs, we use categorical skill level embeddings for two reasons. First, player behavior and decision-making in chess are not linearly related to their rating. Categorical embeddings allow for capturing complex, non-linear relationships between player strength and their moves. They can encode nuanced differences in play style and strategy that are not directly proportional to player ratings. Second, Generalization across similar skill levels: Players within a certain skill level may exhibit similar playing styles, strategies, and common mistakes. Categorical embeddings group players into these ranges, helping the model

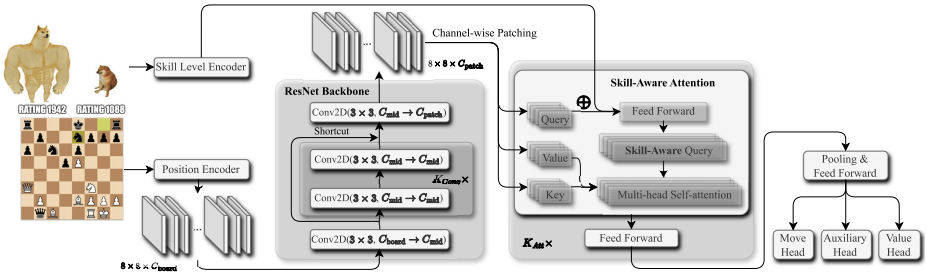

Figure 1: Overview of the Maia-2 model architecture.

to better generalize across players with similar strengths, as opposed to treating each rating as a numerical input.

Let $\mathbf{E} \in \mathbb{R}^{|\mathbf{E}| \times d_s}$ be the matrix of player rating embeddings, where each row corresponds to the embedding of a skill level with dimension d_s: $\mathbf{E} = [\mathbf{e}_{(0,1000]}, \mathbf{e}_{(1000,1100]}, ..., \mathbf{e}_{(2000,+\infty)}]^\top$. Given the skill levels a and o of an active player (i.e. the player to move) and the opponent player, we look up the embedding matrix $\mathbf{E}$ by rows to map the skill levels to active and opponent skill embeddings: $\mathbf{e}_a = \mathbf{E}[a], \mathbf{e}_o = \mathbf{E}[o]$.

Note that previous work [1, 26] uses completely independent models for human-AI alignment at different skill levels—e.g. decisions by 1100-rated chess players are encoded in one model and decisions by 1500-rated players are encoded in a separate model. Further, these models ignore opponent skill level, meaning that predictions cannot vary as a function of opponent strength. However, the active player's decisions may be significantly affected by the opponent's skill level in certain types of situations, or even in general. Players may adjust their strategy based on their perception of the opponent's skill, e.g. a higher-skill opponent might prompt more (or less) cautious play, while against a lower-skill opponent a player may pursue more aggressive tactics. Thus, the interaction between the skill levels of both players is an important component of matching human moves. Unlike existing models that ignore opponent skill level (and actually only consider games in which both players are at the same skill level), we explicitly model not only opponent skill but also the complex interplay between the two players' skill levels, and how it affects human decision-making.

3.2 Position Encoder

Position representation. We use a well-established method [10, 1] to represent each chess position as a multi-channel tensor $P_{\text{input}} \in \mathbb{R}^{C_{\text{board}} \times 8 \times 8}$, which includes channels for each type of chess piece, which color is to move, and states of the position that are not derivable from the position alone (castling rights and en passant), where C_{board} denotes the number of channels. One important departure from previous work is that we only use the current chess position, and not the last few chess positions that occurred in the game (models have typically incorporated the six most recent positions in the game). Many games with perfect information, including chess, can be modeled as alternating Markov games [27, 7], where future states are independent of past states given the current game state. Therefore, the current chess position theoretically encapsulates all the information necessary to make future decisions. Although human decision-making in chess may sometimes subtly depend on the historical lead-up to the current position, these effects are anecdotally small. In exchange, we gain two large practical benefits. First, modeling AI-human move matching in a Markovian way vastly improves training *efficiency* by reducing the computational load via significantly smaller data usage for each decision. Second, it also enhances *flexibility*, enabling our resulting model to make predictions even without historical data, which is particularly advantageous in situations where only the current position is available, like chess training puzzles or any position that didn't necessarily occur in a full game.

Position encoding. To process the position representation P_{input}, we encode P_{input} with the well-established ResNet-based [28] backbone architecture for chess position modeling with K_{Conv} sequentially connected blocks [1]: $P_{\text{encoded}} = \text{Backbone}_{\times K_{\text{Conv}}}(P_{\text{input}}) \in \mathbb{R}^{C_{\text{patch}} \times 8 \times 8}$, where P_{encoded} denotes the encoded position representation of C_{patch} channels. More details about position representation and the backbone architecture can be found in Appendix Section B.

3.3 Bridging Skill Levels and Positions

A central challenge we face is learning how players at different skill levels interact with chess positions differently. How does an expert player evaluate and process a chess position to come up with a move, and how does this differ from a novice? The relationship between positions and skill levels is complicated by the non-linearity in how players of various skill levels interpret and react to chess positions. This complexity presents a significant challenge in human move prediction using a unified model for diverse skill levels. To bridge skill levels and positions—decision-makers and decisions—we propose *skill-aware attention with channel-wise patching*.

Channel-wise patching. In contrast to the area-wise patching approach in Vision Transformers (ViTs) [29], we employ channel-wise patching. Each channel is flattened and linearly transformed, regarding the number of channels in P_{encoded}, i.e., C_{patch}, as the sequence length: $P_{\text{patched}} = \text{Patching}(P_{\text{encoded}}) \in \mathbb{R}^{C_{\text{patch}} \times 64}$, $P = P_{\text{patched}}\mathbf{W} + \mathbf{b} \in \mathbb{R}^{C_{\text{patch}} \times d_{\text{att}}}$, where $\mathbf{W} \in \mathbb{R}^{64 \times d_{\text{att}}}$ and $\mathbf{b} \in \mathbb{R}^{d_{\text{att}}}$ denote the parameters of the linear projection from the patching dimension to the hidden dimension of the skill-aware attention blocks d_{att}. This is particularly suitable for patching encoded chess positions as inputs to Transformer-like architectures, where channels are essentially feature maps that represent different learned latent concepts. These concepts in feature maps are then interactively selected and aggregated considering skill levels via skill-aware attention.

Skill-aware Attention. Given position representations P_{patched} and skill level representations $\mathbf{e}_a$ and $\mathbf{e}_o$, our proposed skill-aware multi-head self-attention is computed as follows. For each head k, we learn weight matrices $\mathbf{W}_k^Q \in \mathbb{R}^{d_{\text{att}} \times d_h}$, $\mathbf{W}_k^K \in \mathbb{R}^{d_{\text{att}} \times d_h}$, and $\mathbf{W}_k^V \in \mathbb{R}^{d_{\text{att}} \times d_h}$, where d_h denote the dimension of each head. The queries Q_k, keys K_k, and values V_k for each head are computed as: $Q_k = P_{\text{patched}}\mathbf{W}_k^Q$, $K_k = P_{\text{patched}}\mathbf{W}_k^K$, $V_k = P_{\text{patched}}\mathbf{W}_k^V$. In order to fuse player skill levels and chess positions progressively and interactively, we inject skill level embeddings into queries within the multi-head self-attention: $Q_k^* = Q_k + (\mathbf{e}_a \oplus \mathbf{e}_o)\mathbf{W}^*$, where $\mathbf{W}^* \in \mathcal{R}^{2d_s \times d_h}$ denotes the weight matrix for feature transformation to the query space, and $\oplus$ is the concatenation operator. We choose to incorporate skill levels in queries because queries directly influence how attention is distributed across patched channels. Using skill-aware queries Q_k^*, the attention mechanism can adjust its focus to reflect the strategic considerations and positional understanding of players at different skill levels. This adjustment allows Maia-2 to adaptively prioritize features of the positions that are more relevant to the skill levels involved, enhancing the model's contextual sensitivity. The skill-aware scaled dot-product attention for each head is thus defined as: $h_k = \text{softmax}\left(\frac{Q_k^* K_k^T}{\sqrt{d_k}}\right) V_k$. The outputs of all heads $h_1, h_2, \ldots, h_h$ are concatenated and then linearly transformed: $P_{\text{att}} = \sigma((h_1 \oplus h_2 \oplus \ldots \oplus h_h)\mathbf{W}^O)$, where $\mathbf{W}^O \in \mathbb{R}^{hd_h \times d_{\text{att}}}$ denote the weight matrix for multi-head attention and $\sigma(\cdot)$ denotes the activation function. We apply the vanilla ViT's feed-forward network and add & norm components upon P_{att} to obtain the output of each skill-aware attention block P_{out}. In Maia-2, we employ a sequence of skill-aware attention blocks to progressively fuse skill levels and positions. Specifically, the output P_{out} for the previous block is fed into the next block as the input. We denote the final output after K_{Att} blocks as P. This procedure enables the model to refine its understanding and interpretation of the positions with each successive block.

3.4 Model Training

Infusing auxiliary information. To enhance the model's understanding of the game state, we inject auxiliary information as labels, including *legal moves* represented by multi-hot vectors and *human move information*: one-hot vectors of which piece is moved, which piece is captured (if any), the move's originating square, the move's destination square, and whether or not the move will deliver a check. These segments are used as labels for classification, serving a dual purpose: 1) It offers a more granular understanding of human moves by providing detailed context beyond just the move indices produced by the policy head labels, enriching the model's insight of player decisions; and 2) It ensures the model also learns about objective (i.e. chess-specific as opposed to behavioral) knowledge in chess, which is essential for developing a comprehensive understanding of both human moves and the fundamental mechanics of the game.

Table 1: Move prediction accuracy on the *Maia-2 Testset*. *Skilled*, *Advanced*, and *Master* are grouped according to Section 4 and *Avg* denotes macro-averaged results.

	Stockfish			Leela			Maia			Maia-2_{subset}	Maia-2
	3	9	15	1500	2200	3200	1100	1500	1900		
Skilled	36.22	36.00	36.86	40.46	39.79	39.97	51.48	50.79	48.51	51.51	**51.72**
Advanced	38.25	38.78	39.83	44.45	43.97	44.29	49.13	52.61	52.26	53.54	**54.15**
Master	40.71	43.26	44.61	48.69	47.11	47.75	45.85	50.76	53.20	53.16	**53.87**
Avg	38.39	39.35	40.43	44.53	43.62	44.00	48.82	51.39	51.32	52.74	**53.25**

Data balancing and filtering. Chess games between players of significantly different skill levels are relatively rare but help us understand how players of lower skill levels approach games against far stronger opponents and vice versa. While previous work has ignored these games completely, they play a central role in our approach. Since games between players of similar skill levels vastly outnumber more uneven matchups, we use a data balancing strategy to effectively train our unified model for aligning players across all skill levels, in which games between players of different skill levels are over-sampled. Online chess platforms feature a variety of game types, including blitz, rapid, and classical, each representing games played at different time controls (amount of time given to each player for the whole game). We focus on Rapid games, which are medium-length games that lie between the fast-paced decisions of "Blitz" games and the slower, more strategic considerations of "Classical" games. In addition, we follow the procedures in [1] to filter valid positions within each game. More details about data balancing and filtering are available in Appendix B.

Training objectives. With the fused skill level and position representation P as input, we construct the policy head on top to predict human moves, which is optimized using cross-entropy loss with one-hot labels representing the recorded human move. We also build the auxiliary information head to infuse additional knowledge into Maia-2 as introduced in Section 3.4. This head is trained using bit-wise binary cross-entropy loss with multi-hot labels. Finally, following previous work [1, 25] we include a value head to predict the game outcome as a regression task, where the labels 1, 0, -1 denote winning, drawing, and losing, respectively. The training objectives of these heads are balanced to contribute equally to Maia-2 model optimization. Hyperparameter settings used for Maia-2 training can be found in Appendix Table 5.

4 Results

We empirically evaluate Maia-2 along two key dimensions: move prediction *accuracy*, how well it can predict human moves at varying skill levels, and move prediction *coherence*, how aligned its predictions are across skill levels. We train Maia-2 on Lichess games played between Jan 2013 and Nov 2023, with the exception of December 2019, since that is the month used for testing in the original Maia paper (and we also test on this month for consistency) [1]. After game filtering and balancing, we end up with a training set of 169M games (9.1B positions). We also train Maia-2_{subset} with identical model architecture and training configurations as Maia-2, except it only has access to the same training data that Maia had for fair comparisons. Dataset statistics are reported in Appendix Tables 7, 9, and 10. We compare Maia-2 with Stockfish [6], the strongest chess engine, Leela, an open-source counterpart to AlphaZero [10]. and Maia [1], the state-of-the-art model for human-like chess play. Maia is actually a set of 9 separate models, each trained on a different set of players at different skill levels from 1100 to 1900. We use the benchmarking *Maia Testset* [1] for performance comparisons where both players have identical skill levels. We report the results on *Maia Testset* by grouping players into three categories: *Skilled* (Rapid rating up to 1600, which slightly exceeds the initial rating of 1500), *Advanced* (Rapid rating between 1600 and 2000), and *Master* (Rapid rating over 2000). In addition, we aim to evaluate move prediction across diverse skill combinations with the *Cross-skill Testset* constructed from Dec 2023 games. Finally, we construct *Grounded Testset* with 450,000 positions that has recorded Stockfish evaluations, which can serve as grounded facts to measure move quality. Statistics of datasets are summarized in Appendix Table 8.

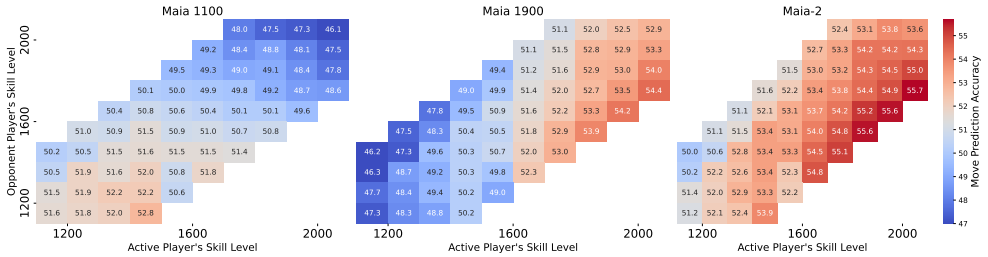

Figure 2: Move prediction accuracy across diverse skill levels. Colors represent performance, with warmer tones indicating higher accuracy.

4.1 Move Prediction Accuracy

Maia-2. In Table 1, we show the top-1 move prediction accuracy of all models across all groups of players on the *Maia-1 Testset*. Maia-2 demonstrates strong and consistent performance across all skill levels, surpassing all baselines. Specifically, despite Maia-1 models being specifically trained to mimic chess moves by players at specific skill levels, Maia-2 emerges as a unified one-for-all model that is consistently effective across the entire spectrum of chess skills. The largest improvement is on *Advanced* players, where Maia-2 gains 1.5 percentage points over the nearest competitor (Maia 1500). When averaging across skill levels, Maia-2 outperforms all other models by almost 2 full percentage points in overall accuracy. Note that the ceiling accuracy of human move prediction is far below 100% given the randomness and diversity of human decisions—even the *same* player won't always make the same decision when faced with the same position. Our 2 percentage point gain is substantial considering that the difference between Maia-1 and Leela, the previous state-of-the-art model for this task and a traditional chess engine not trained for this task at all, is only 6 percentage points. Furthermore, Maia-1 is essentially a mixture of 9 experts targeting the specific players' skill level, where each expert has 10.3M parameters. Regarding the routing function to select the best-performing expert as a nonparameterized function, Maia-1 has 92M parameters in total. Maia-2, on the other hand, is a one-for-all model with 23.3M parameters under our default settings. Therefore, Maia-2 achieves better human move prediction accuracy with even much fewer trainable parameters.

Baseline models. Both Maia-2 and Maia-1 significantly outperform Stockfish and Leela, typically by 5–15 percentage points. Note that Stockfish and Leela aim to play optimal chess (as most humans do too), and only "predict" human moves when their approximations to optimality happen to overlap with those of human players. However, we compare to these traditional chess engines because besides Maia-1, there are still the default method of creating "human-like" AI agents. The accuracy gap between Maia-1 architectures and traditional chess engines demonstrates the necessity of developing specialized models to mimic human chess moves.

Maia-2$_{\text{subset}}$. Maia-2 differs from Maia-1 in two main ways: it has a different architecture and it has access to more training data. To control for the difference in training data and isolate the effects of our architecture, we create Maia-2$_{\text{subset}}$ which has access to the exact same training data that Maia-1 was developed with. Comparing the two, we see that Maia-2$_{\text{subset}}$ matches or outperforms all baselines and alternate models. Recall that Maia-2 and Maia-2$_{\text{subset}}$ don't have the recent history passed as input to them, yet still achieve state-of-the-art results. It is important to note that each Maia-1 model is specifically trained for its respective skill level, relying solely on games where the active and opponent skill levels match for its training data. On the contrary, the unified modeling approach with skill-aware attention of Maia-2$_{\text{subset}}$ allows it to utilize a broader spectrum of games, featuring a variety of skill-level pairings, for training purposes. Consequently, while both Maia-1 and Maia-2$_{\text{subset}}$ draw from the same source dataset, Maia-2$_{\text{subset}}$ can leverage a significantly larger portion of this data for its training, improving its learning and predictive capabilities. The improvement from Maia-2$_{\text{subset}}$ to Maia-2 underscores the importance of extensive training with vast datasets. A broader range of games provides Maia-2 with access to more comprehensive and nuanced patterns in human chess moves. Using Maia-2$_{\text{subset}}$ as a comparison, we can determine the relative contributions of model architecture and training data to Maia-2's 1.9 percentage point gap over its nearest rival (Maia 1500). This calculation suggests that 73% of the increase in performance is due to the architecture improvements and 27% is due to increased training data.

Table 2: Move prediction perplexity on the *Grounded Testset*.

	Skilled	Advanced	Master	Avg
Maia 1100	5.05	5.56	6.01	5.54
Maia 1500	4.98	5.31	5.50	5.26
Maia 1900	4.44	4.71	4.86	4.67
Maia-2	**4.30**	**3.90**	**4.02**	**4.07**

Table 3: Ablation study results, where Maia-2_{subset} is compared with versions without skill-aware attention ("w/o Att") and without infusing auxiliary information ("w/o Aux").

	Skilled	Advanced	Master	Avg
w/o Att	50.63	52.89	51.73	51.75
w/o Aux	50.96	52.98	52.34	52.09
Maia-2_{subset}	**51.51**	**53.54**	**53.16**	**52.74**

Move prediction perplexity. While top-1 accuracy gains are important, they may overshadow larger improvements in prediction quality. To account for this, we also measure the perplexity of move predictions , which reflects the model's confidence in its predictions. A lower perplexity indicates the model is more confident and accurate in human move prediction, as it corresponds to a higher likelihood of the correct human move. As shown in Table 2, Maia-2 consistently yields substantially lower perplexity in all groups of skill levels compared to Maia-1. In particular, Maia-2 significantly outperforms Maia-1 in *Advanced* and *Master* moves with relatively large margins, demonstrating the effectiveness of our unified modeling approach across diverse skill levels.

Adaptive move predictions. We now evaluate Maia-2's ability to predict compare across diverse skill combinations using the *Cross-skill Testset*. As shown in Figure 2, Maia-2 consistently outperforms both Maia 1100 and Maia 1900 in almost all combinations of active player and opponent player skill levels. In particular, although Maia 1100 and Maia 1900 demonstrate competent performance within their respective domains of expertise, their predictive accuracy decreases substantially outside of these targeted skill levels. This is because Maia-1 models are static and cannot respond to varied skill levels and adjust their predictions accordingly. In contrast, our proposed unified modeling approach with skill-aware attention enables Maia-2 to adapt its predictions to account for the skill levels of both the active player and the opponent player, so that varying skill level configurations correspondingly can result in better aligned human move predictions. More results on comparisons with other Maia-1 versions can be found in Figure 6 in the Appendix.

Move quality. One of our key motivations for creating a unified model of human chess behavior is to guide the development of future algorithmic learning tools. As such, understanding the mistakes that people make is of fundamental interest. Can Maia-2 predict mistakes better than Maia-1? Figure 10 in the Appendix shows the move prediction accuracy on the *Grounded Testset* as a function of move quality, measured by win-rate loss, which is calculated following the same procedures as prior studies [1, 25]. All models generally decrease in their ability to predict worse moves, since humans are generally trying to avoid mistakes, and high-quality moves are more certain whereas lower-quality moves can be more random and thus hard to predict. Nevertheless, Maia-2 outperforms all versions of Maia-1 across most of the move quality range, demonstrating the effectiveness of our unified modeling approach for human move prediction.

We are also interested in how certain the models are about their predictions of various move qualities. Figure 3.(A)(top) shows the probabilities that Maia-2 (x-axis) and Maia-1 (y-axis) attribute to the moves people actually played in *Grounded Testset*. The concentration of points in the lower right quadrant (closer to P(Maia-2) = 1 and P(Maia 1900) = 0) suggests that Maia-2 assigns a higher probability to the true move than Maia-1 does, indicating superior predictive performance. Conversely, the less dense upper left quadrant indicates fewer instances where Maia 1900 outperforms Maia-2. Remarkably, while this consistently occurs across all move qualities, the distinction is more pronounced for *Blunders* and *Errors* compared to *Optimal* moves. Additionally, the bottom row of Figure 3.(A) shows the log odds ratio between $P(x, y)$ and $P(y, x)$ in the top row. The abundance of blue points below the diagonal indicates that Maia-2 is almost always more confident in the correct move than Maia-1 is, indicating an across-the-board improvement in move prediction. Maia-2 offers superior and more confident prediction across diverse move qualities. We also conduct an ablation study as shown in Table 3, we point the readers to Appendix A for more information.

4.2 Move Prediction Coherence

Maia-2's accuracy across the spectrum of human skill is certainly desirable, but perhaps an even more important dimension is prediction *coherence* as skill varies. A central drawback of Maia-1 is that it

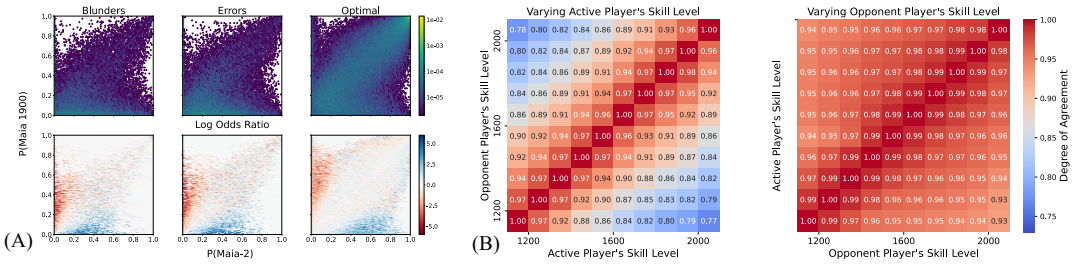

Figure 3: (A). (Top) Joint probability assigned to human moves played by Maia-2 (x) and Maia 1900 (y), split by move quality. Blunders (left) reduce the expected win-rate by $\geq 10\%$, Errors (middle) by 5–10%, and Optimal (right) by $\leq 0\%$. (Bottom) Log odds ratio of $p(x, y)$ and $p(y, x)$ from top. (B). Move prediction agreement as (left) active player and (right) opponent player skill are varied. All cells are evaluated on the same set of positions but with altered skill level configurations.

Table 4: Percentage of monotonic and transitional positions.

	%Monotonic			%Transitional		
	Skilled	Advanced	Master	Skilled	Advanced	Master
Maia-1	1.61	1.42	1.14	13.34	18.14	20.48
Maia-2	27.61	28.51	26.38	22.59	23.39	21.72

models players at different skill levels independently from each other, which results in particularly volatile predictions: the same position might elicit very different predicted behavior from models of adjacent skill levels. This is problematic because we know from personal experience that this type of volatility is rare: players don't change that much as they improve. This limits Maia's ability to perform well in downstream tasks such as serving as a teaching aid, as its understanding of one skill level bears little resemblance to its understanding of the next. In reality, players move from one skill level to another by making small, consistent adjustments. Does Maia-2 reflect this behavioral coherence?

Prediction smoothness. We measure the coherence of Maia-2's predictions by testing for smoothness features in its entire set of predictions. Call a model's treatment of a position *monotonic* if the predicted probability of the correct move increases with skill monotonically. In the *Grounded Testset* of 100K positions, we find that Maia-1 only treats 1% of them monotonically. In stark contrast, however, Maia-2 treats 27% of them monotonically, clearly demonstrating that Maia-2 is much more coherent. Similarly, call a model's treatment of a position *transitional* if it predicts a suboptimal move for some prefix of skills and then transitions to an optimal move for all subsequent skill levels. Again, Maia-2 treats substantially more positions transitionally—around 22% of them compared with 17% for Maia.

It's important to note that Maia-2 is deliberately designed to encourage coherence across skill levels without rigidly enforcing it. Our objective is not to impose coherence as a hard constraint, which might obscure legitimate differences in player behavior between skill levels, but to create a model architecture that naturally encourages coherence where the data supports it.

Move prediction agreement. As a first test, we measure move prediction coherence as we vary active player skill and opponent player skill in Maia-2. The results shown in Figure 3.(B) reveal several trends. First, increasingly varying either the active or opponent rating results in lower agreement, suggesting that Maia-2 smoothly varies its predictions with skill. Second, comparing the two heatmaps reveals that Maia-2 has clearly learned that varying one's own skill has much larger effects than varying the opponent's—changing one's own skill against a fixed opponent can change the decision up to 22% of the time, but changing the opponent's skill while fixing our own skill will only change the decision up to 6% of the time. This is intuitive, as players must change their decisions in order to play at a higher level, while in theory one's opponent shouldn't affect one's decision. Of course, humans are not optimal agents and sometimes take their opponent's skill level into account when deciding on a move—willfully or not—which is reflected in our results.

Chess concept understanding. Human chess players of varying strengths differ in their ability to recognize important features and patterns on the board, e.g., stronger players are adept at discerning

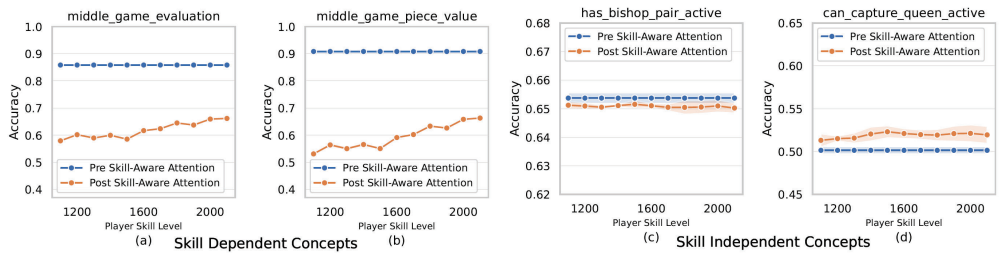

Figure 4: Maia-2's chess concept recognition as a function of skill level, as measured by linear activation probes right before (blue) and after (orange) skill-aware attention. (a) Stockfish overall board evaluation for middle-game positions. (b) Stockfish evaluation of middle-game bonuses and penalties to pieces for white. (c) Does the active player own two bishops? (d) Can the active player capture the opponent's queen?

subtle nuances. We now turn our focus to a critical question: does Maia-2 vary in its ability to capture human chess concepts when given different skill levels? Following the chess concepts probing strategy for AlphaZero [22], we show how Maia-2's grasp of various concepts varies with skill. The left two plots in Figure 4 show concepts for which Maia-2 clearly distinguishes between skill levels, with higher-skill players paying more attention to them than lower-skill players. These are general board evaluations as given by Stockfish [6], or aggregate piece values. Note that pre-skill-aware attention is always flat because by construction it cannot vary with skill, since skill-aware attention has not been applied yet. The two plots on the right depict concepts that live closer to fundamental chess rules, and as such are less dependent on player skill. For skill-dependent concepts, the figures reveal an increasing trend in mastery level after skill-aware attention, aligning with the increase in dedicated skill levels. Meanwhile, the model's mastery level decreases after passing through the skill-aware attention modules, potentially adjusting for the imperfections of human players. Conversely, the skill-aware attention blocks are not responsive to skill-independent concepts.

5 Discussion

Human Study. In addition to human move matching, we also consider engagement, another dimension of human study. In particular, we implement a randomized experiment on Lichess: human players challenge our bots, and we randomize whether players play against Maia-1 or Maia-2. Our result is that our higher move-matching and our vastly improved coherence, across all skill levels, come at no cost to human subject engagement, and in fact slightly increase engagement: players rematch Maia-2 almost 1 percentage point more than Maia-1 (41.2% vs. 40.3%). Although engagement is not our main objective, this is further promising evidence that we have achieved a more human-aligned model that coherently captures human style across different skill levels.

Ethical Considerations. We believe Maia-2 poses limited risk while offering large potential benefits. Our data is highly aggregated, with almost 1 billion games being used for training, and chess as a domain is generally low-risk. Meanwhile, helping people improve in chess could lead to increased cognitive skills, confidence boosts, and help with general life satisfaction. Our vision is for Maia-2 to power AI partners and training aids; it cannot currently replace skilled human tutors and coaches.

Limitation. Our work has limitations. First, we are excited by the applications that Maia-2 will enable, such as more relatable AI partners and AI-powered learning aids, the development of which is out of scope for the current work. Maia-2 does not yet incorporate search, although previous work has demonstrated that with proper regularization it can help improve move prediction performance [26]. Relatedly, we group the strongest players in a single bucket, although modeling the very best players in the world remains difficult due to the complexity and depth of their moves.

Acknowledgements

We thank the anonymous reviewers for helpful comments. This research was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC), a Microsoft Research Award, a Simons Collaboration grant, and a grant from the MacArthur Foundation.

References

[1] Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Aligning superhuman ai with human behavior: Chess as a model system. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1677–1687, 2020.

[2] Feng-hsiung Hsu. Ibm's deep blue chess grandmaster chips. *IEEE micro*, 19(2):70–81, 1999.

[3] William Clark, Jan Golinski, and Simon Schaffer. *The sciences in enlightened Europe*. University of Chicago Press, 1999.

[4] Frederic Friedel. Reconstructing turing's "paper machine".

[5] Claude E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:623–656, 1948.

[6] Tord Romstad, Marco Costalba, Joona Kiiski, and et al. Stockfish. stockfishchess.org, 2023. Accessed: 2024-01-05.

[7] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[8] Xidong Feng, Yicheng Luo, Ziyan Wang, Hongrui Tang, Mengyue Yang, Kun Shao, David Mguni, Yali Du, and Jun Wang. Chessgpt: Bridging policy learning and language modeling. *arXiv preprint arXiv:2306.09200*, 2023.

[9] Feng-Hsiung Hsu. *Behind Deep Blue: Building the computer that defeated the world chess champion*. Princeton University Press, 2002.

[10] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv*, 2017.

[11] Tom Zahavy, Vivek Veeriah, Shaobo Hou, Kevin Waugh, Matthew Lai, Edouard Leurent, Nenad Tomasev, Lisa Schut, Demis Hassabis, and Satinder Singh. Diversifying ai: Towards creative chess with alphazero. *arXiv preprint arXiv:2308.09175*, 2023.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[13] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[14] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[15] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[16] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

[17] Max Hager. LLMChess.

[18] Nicholas Carlini. carlini/chess-llm.

[19] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[20] Andrew Lee, David Wu, Emily Dinan, and Mike Lewis. Improving chess commentaries by combining language models with symbolic reasoning engines. *arXiv preprint arXiv:2212.08195*, 2022.

[21] David Noever, Matt Ciolino, and Josh Kalin. The chess transformer: Mastering play using generative language models. *arXiv preprint arXiv:2008.04057*, 2020.

[22] Thomas McGrath, Andrei Kapishnikov, Nenad Tomašev, Adam Pearce, Martin Wattenberg, Demis Hassabis, Been Kim, Ulrich Paquet, and Vladimir Kramnik. Acquisition of chess knowledge in alphazero. *Proceedings of the National Academy of Sciences*, 119(47):e2206625119, 2022.

[23] Lisa Schut, Nenad Tomasev, Tom McGrath, Demis Hassabis, Ulrich Paquet, and Been Kim. Bridging the human-ai knowledge gap: Concept discovery and transfer in alphazero. *arXiv preprint arXiv:2310.16410*, 2023.

[24] Reid McIlroy-Young, Yu Wang, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Detecting individual decision-making style: Exploring behavioral stylometry in chess. *Advances in Neural Information Processing Systems*, 34:24482–24497, 2021.

[25] Reid McIlroy-Young, Russell Wang, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Learning models of individual behavior in chess. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1253–1263, 2022.

[26] Athul Paul Jacob, David J Wu, Gabriele Farina, Adam Lerer, Hengyuan Hu, Anton Bakhtin, Jacob Andreas, and Noam Brown. Modeling strong and human-like gameplay with kl-regularized search. In *International Conference on Machine Learning*, pages 9695–9728. PMLR, 2022.

[27] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

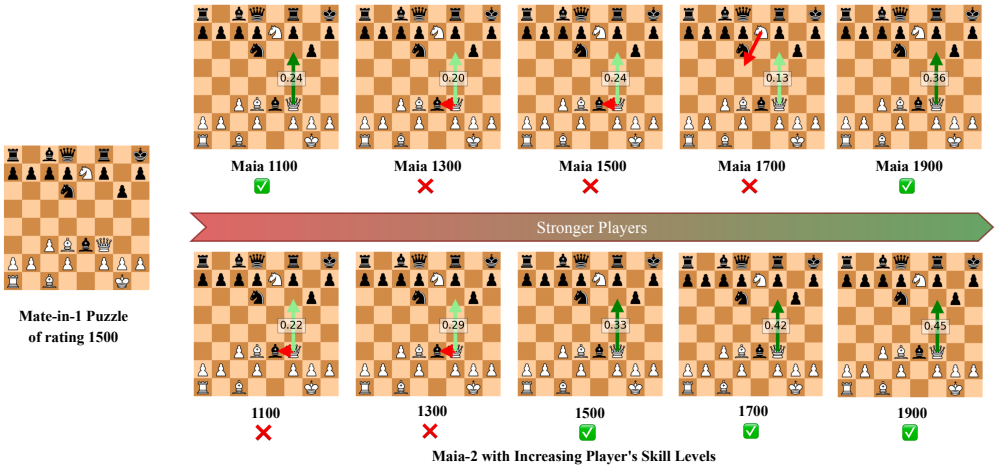

Figure 5: Maia-2 and Maia-1 solving a Mate-in-1 chess puzzle of rating 1500. Green arrows represent correct move predictions, while red arrows indicate incorrect predictions. The darkness of the green color correlates with the model's confidence, with darker arrows denoting a higher probability of making the correct move.

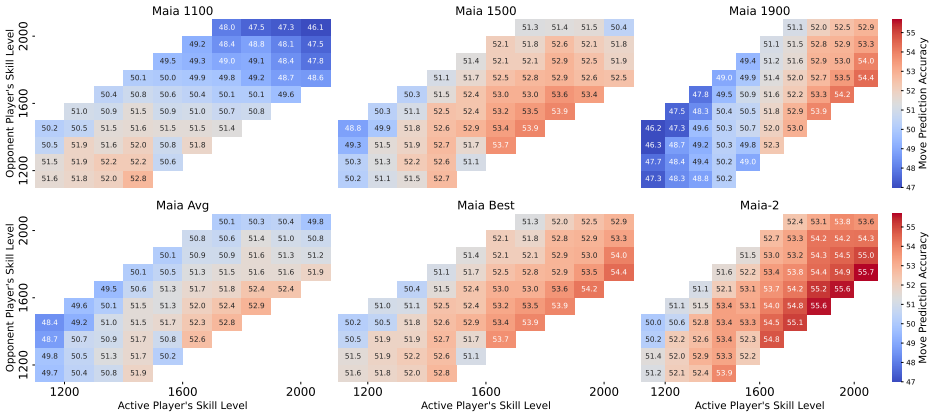

Figure 6: Move prediction accuracy across diverse skill levels. Colors represent performance, with warmer tones indicating higher accuracy. Missing skill combinations were too rare to be statistically reliable.

A More Experimental Results

Case study: Smoothness. We evaluate the smoothness of Maia-1 and Maia-2 by a case study in puzzle solving: a Mate-in-1 puzzle of 1500 skill level is presented to both Maia-1 and Maia-2, where smoothness can be evaluated by checking whether the predictions are *monotonic* and *transitional* as skill level increases. Call a model's treatment of a position as *monotonic* if the predicted probability of the correct move increases with skill monotonically, we can observe from Figure 5 that the Maia-2 predicted probabilities of the best move (in green arrows) increase monotonically from 0.22 to 0.45 as the skill levels rise from 1100 to 1900, while Maia-1 predictions are rather turbulent. Similarly, we call a model's treatment of a position *transitional* if it predicts a suboptimal move for some prefix of skills and then transitions to an optimal move for all subsequent skill levels. As shown in Figure 5, Maia-2 can mimic weaker players to whom the puzzle is hard to solve, while stronger Maia-2 with skill level configured above or equal to 1500 can successfully solve the puzzle. However, Maia 1100 surprisingly solved the puzzle, while the stronger Maia-1 models, e.g., Maia 1700 failed to make the optimal move. Therefore, in the considered case, as opposed to Maia-1, Maia-2 yields smooth predictions provided that its treatment of this position is *monotonic* and *transitional*.

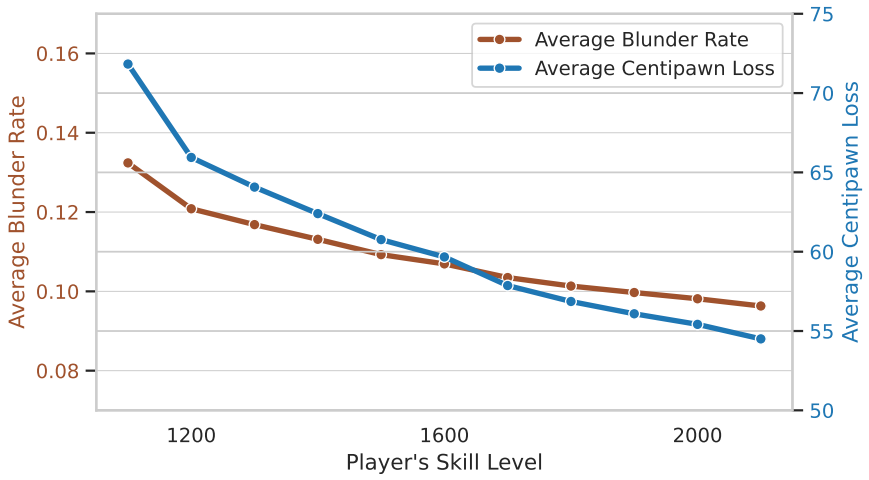

Figure 7: Quality of predicted moves quantified by blunder rate and centipawn Loss.

Quality of predicted moves. Given the same chess position but increasing skill levels, a coherent, skill-aware human move prediction model should make progressively higher-quality moves. We measure the average centipawn loss (a standard move quality metric, the lower the better) and the average blunder rate (fraction of times an egregious mistake is predicted) in positions randomly sampled from December 2023 games. The centipawn loss is determined by comparing the moves predicted by Maia-2 against the top move as evaluated by Stockfish at depth 20 (human grandmaster-level play), while blunders are classified as moves resulting in a win-rate loss of 10% or more. As shown in Figure 7, both the centipawn loss and the blunder rate exhibit a smooth, monotonic decrease as player skill rises, demonstrating Maia-2's capability of adjusting its predictions coherently to align with the increasingly skilled players.

Value head. As a proxy of model evaluation given a board position, we train the model value head which is potentially significant for a wide variety of downstream tasks. The value head is trained as a regression task from -1 to 1 indicating from losing to winning positions, and finally normalized to a continuous value between 0 and 1 similar to AlphaZero. The correct label for value head is the actual game results. To check the evaluation quality of our model value head, we calibrate the value head results with actual game outcomes, and generate a quantile-quantile plot in Figure 8 where the win probability is discretized uniformly into 100 bins from 0 to 1.

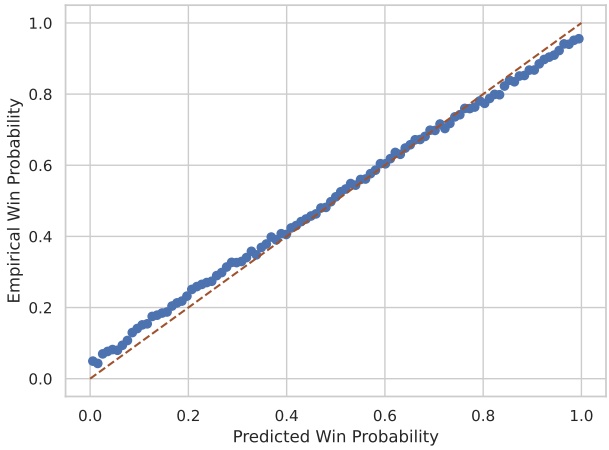

Figure 8: Value head Q-Q plot.

Ablation study. To understand which components of our architecture are most responsible for the performance gains, we conduct an ablation study on the *Maia Testset*. We train a version of Maia-2_{subset} using a naive method of incorporating encoded skill levels without the proposed skill-aware attention module ("w/o Att"). In this model, the skill level encodings $\mathbf{e}_a$ and $\mathbf{e}o$ are directly concatenated with flattened P_{encoded}, without bridging them with the skill-aware attention (skipping Section 3.3), and directly connected to prediction heads. As shown in Table 3, Maia-2_{subset} consistently performs better with skill-aware attention, demonstrating the necessity of modeling the complexity and non-linearity in player's skill development in sophisticated ways and the effectiveness of our proposed unified modeling approach with skill-aware attention to model such nuances. We also train a version of Maia-2_{subset} without the auxiliary information head ("w/o Aux"). These results show that infusing auxiliary information as labels during model training also results in a significant performance improvement, although not as dramatically as skill-aware attention.

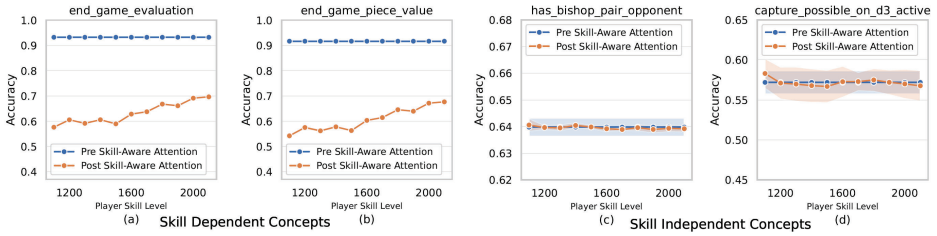

Figure 9: Maia-2's chess concept recognition as a function of skill level, as measured by linear activation probes right before (blue) and after (orange) skill-aware attention. (a) Stockfish overall board evaluation for end-game positions. (b) Stockfish evaluation of end-game bonuses and penalties to pieces for white. (c) Does the opponent player own two bishops? (d) Is capture possible on square d3 for the active player?

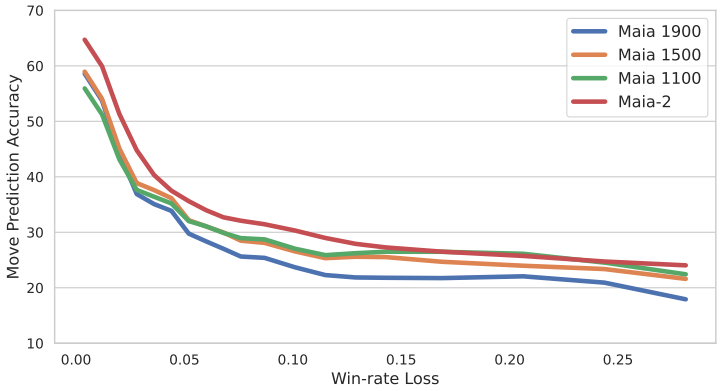

Figure 10: Move prediction accuracy as a function of move quality, quantified by win-rate loss.

Accuracy as a function of move quality. Figure 10 shows the move prediction accuracy on the *Grounded Testset* as a function of move quality, measured by win-rate loss, which is calculated following the same procedures as prior studies [1, 25]. All models generally decrease in their ability to predict worse moves, since humans are generally trying to avoid mistakes, and high-quality moves are more certain whereas lower-quality moves can be more random and thus hard to predict. Nevertheless, Maia-2 outperforms all versions of Maia across most of the move quality range, demonstrating the effectiveness of our unified modeling approach for human move prediction. Maia-2's overall gains are not constrained to any specific move quality type, but are spread across the entire range.

Table 5: Hyperparameter Settings.

#Games per chunk N_{chunk}	20000
#Maximum games per skill level N_{range}	20
Initial learning rate	$1e^{-4}$
Weight decay	$1e^{-5}$
Batch size (positions)	8192
Minimum move ply	10
Maximum move ply	300
Remaining seconds threshold	30
#Backbone blocks K_{Conv}	12
#Attention block K_{Att}	2
#Input channels C_{input}	18
#Intermediate channels C_{mid}	256
#Encoded channels C_{patch}	8
Skill level embedding dimension d_s	128
Attention head dimension d_h	64
Attention intermediate dimension d_{att}	1024
#Attention heads h	16

Table 6: Statistics of the *Maia-1 Testset*.

	#Positions	Rating Range
Total	106,740	-
Skilled	56,812	< 1599
Advanced	41,747	1600 - 1999
Master	8,181	$\geq$ 2000

B Reproductibility

Implementation details. To maintain a consistent perspective from both sides of players, we implemented board flipping to train and test Maia-2; that is, positions with black to move were mirrored such that all analyses could be conducted from the white side's viewpoint. We further refined our dataset through game and position filtering, selecting only rapid games from Lichess with available clock information and disregarding the initial 10 plys of each game as well as positions where either player had less than thirty seconds remaining. The filtration is significant for eliminating the noise introduced by rushed decisions under time constraints, which could skew the true representation of a player's skill. The choice of exclusive rapid games is also informed by the distinct rating systems across different game types, which necessitates the separation of data to maintain rating consistency. We report all hyperparameters involved in training Maia-2 in Table 5. It took approximately 13 days to train Maia-2 with 2×A100 (80G) GPUs under our default settings.

Data Balancing. Chess games between players of significantly different skill levels are relatively rare but help us understand how players of lower skill levels approach games against far stronger opponents and vice versa. While previous work has ignored these games completely, they play a central role in our approach. Since games between players of similar skill levels vastly outnumber more uneven matchups, we use a data balancing strategy to effectively train our unified model for aligning players across all skill levels, in which games between players of different skill levels are

Table 7: Statistics of training datasets.

	Maia-2_{subset}	Maia-2
#Games Consumed	930.19M	5.14B
#Games Trained	21.31M	168.93M
#Positions Trained	1.18B	9.15B

over-sampled. To be precise, we pre-process the data in chunks of N_{chunk} games each. We then scan each data chunk to find games satisfying various (active player skill, opponent skill) combinations. Each skill combination can include at most N_{range} games. We continue scanning the data chunk until all the skill combinations have N_{range} games or the data chunk is fully consumed. Note that the higher the balancing factor $\frac{N_{\text{range}}}{N_{\text{chunk}}}$, the less likely it is that rare skill combinations will fully reach N_{range}, which will lead to less balanced data overall. On the other hand, if the balancing factor is too small, fewer games will be selected from each data chunk, which is data-inefficient. We choose a fair compromise between data efficiency and balanced data so that our training data encompasses a broad spectrum of skill levels without biasing excessively towards the more frequently-occurring equal-skill matchups.

Data Filtering. Online chess platforms feature a variety of game types, including blitz, rapid, and classical, each representing games played at different time controls (amount of time given to each player for the whole game). We use data from Lichess, a well-known large open-source chess platform, and its open database. In Lichess, since each game type is given a separate rating, ratings across different game types are not comparable (e.g. a rating of 1800 in "Rapid" is significantly weaker than a rating of 1800 in "Blitz" on Lichess). Previous work [1, 25] mixes player ratings across these game types together for training and evaluation. Instead of mixing data across game types, we focus on Rapid games only, which are medium-length games that lie between the fast-paced decisions of "Blitz" games and the slower, more strategic considerations of "Classical" games. "Blitz" and "Bullet" games, characterized by their quick pace, are composed of many decisions made under time pressure, introducing randomness that may not accurately reflect player intentions and skills. In contrast, Classical games are played less frequently, leading to data scarcity. Rapid chess is an ideal compromise between quality of play and quantity of data. In addition, we follow the procedures in [1] to filter valid positions within each game.

Chess position representation. We follow the well-established prior works [10, 1] to represent chess positions as multi-channel 8×8 matrices, including:

- Piece Representation: The first 12 channels categorize the board's pieces by type and color, with one channel each for white and black Pawns, Knights, Bishops, Rooks, Queens, and Kings. A cell is marked 1 to denote the presence of a piece in the corresponding location, and 0 otherwise.
- Player's Turn: A single channel (the 13th) indicates the current player's turn, filled entirely with 1s for white and 0s for black, providing the model with context on whose move is being evaluated.
- Castling Rights: Four channels (14th to 17th) encode the castling rights for both players, with the entire channel set to 1 if the right is available or 0 otherwise.
- En Passant Target: The final channel (18th) marks the square available for en passant capture, if any, with 1 and 0s elsewhere.

Chess concepts probing. Given a board position, we vary the skill level injected to Maia-2 to extract the $P_{\text{encoded}} \in \mathbb{R}^{C_{\text{patch}} \times 8 \times 8}$ as the learned respresentation of the ResNet-based backbone served as the control group which remain constant to varying skill ratings, and the output hidden states $P \in \mathbb{R}^{C_{\text{patch}} \times d_{\text{att}}}$ after skill-aware attention directly connected to model heads. We randomly pick 500,000 positions from Lichess December 2023 database and calculate Stockfish built-in or custom implemented concepts using the Forsyth-Edwards Notation of the positions.

We carefully choose P_{encoded} and P as internal representations for the reason that P_{encoded} is the comprehensive understanding of chess board for the backbone network right before skill-aware attention, and P is the final hidden states that directly influence model decision and evaluation. Following the work of Alphazero concept probing [22], for chess concepts with continuous values we train Lasso regressors with coefficient for $\mathcal{L}^1$-regularization selected by 5-fold cross-validation, and for binary concepts we train logistic regressors with downsampling. During evaluation on unbalanced test set, we measure the performance of continuous-valued probes with the coefficient of determination r^2, and score the binary-valued probes with the Macro F1.

Table 8: Statistics of the *Cross-skill Testset*.

Opponent	Active								
	1100 - 1199	1200 - 1299	1300 - 1399	1400 - 1499	1500 - 1599	1600 - 1699	1700 - 1799	1800 - 1899	≥2000
≥2000	-	-	-	-	-	91,264	100,000	100,000	100,000
1800 - 1899	-	-	-	-	83,337	99,931	100,000	100,000	100,000
1700 - 1799	-	-	-	91,054	100,000	100,000	100,000	100,000	100,000
1600 - 1699	-	-	81,386	99,998	100,000	100,000	100,000	99,933	91,610
1500 - 1599	-	82,059	99,500	100,000	100,000	100,000	99,974	83,650	-
1400 - 1499	89,666	100,000	100,000	100,000	100,000	100,000	91,291	-	-
1300 - 1399	97,602	100,000	100,000	100,000	99,544	81,692	-	-	-
1200 - 1299	100,000	100,000	100,000	100,000	82,230	-	-	-	-
1100 - 1199	100,000	100,000	97,635	89,643	-	-	-	-	-

Table 9: Number of games in the balanced dataset used for training Maia-2_{subset}.

Black	White										
	<1100	1100 - 1199	1200 - 1299	1300 - 1399	1400 - 1499	1500 - 1599	1600 - 1699	1700 - 1799	1800 - 1899	1900 - 1999	>=2000
<1100	583,437	-	-	-	-	-	-	-	-	-	-
1100 - 1199	583,403	583,433	-	-	-	-	-	-	-	-	-
1200 - 1299	427,160	583,402	583,443	-	-	-	-	-	-	-	-
1300 - 1399	176,930	408,819	583,421	583,443	-	-	-	-	-	-	-
1400 - 1499	113,683	204,425	499,185	583,433	583,445	-	-	-	-	-	-
1500 - 1599	101,756	139,741	307,446	559,722	583,440	583,439	-	-	-	-	-
1600 - 1699	57,381	85,483	174,417	384,670	566,851	583,436	583,435	-	-	-	-
1700 - 1799	32,621	47,671	100,841	188,967	359,056	571,666	583,421	583,428	-	-	-
1800 - 1899	16,656	26,195	52,043	105,095	178,957	348,979	557,921	583,415	583,422	-	-
1900 - 1999	8,032	11,736	23,800	48,965	84,808	150,182	272,451	529,071	583,388	582,715	-
>2000	5,927	7,953	14,825	28,929	49,237	88,317	134,432	280,452	531,162	582,993	582,542

Table 10: Number of games in the balanced dataset used for training Maia-2.

Black	White										
	<1100	1100 - 1199	1200 - 1299	1300 - 1399	1400 - 1499	1500 - 1599	1600 - 1699	1700 - 1799	1800 - 1899	1900 - 1999	>=2000
<1100	4,698,087	-	-	-	-	-	-	-	-	-	-
1100 - 1199	4,697,937	4,697,979	-	-	-	-	-	-	-	-	-
1200 - 1299	3,165,114	4,697,972	4,698,076	-	-	-	-	-	-	-	-
1300 - 1399	2,015,110	2,561,344	4,698,015	4,698,073	-	-	-	-	-	-	-
1400 - 1499	1,509,456	1,549,218	3,008,520	4,698,033	4,698,093	-	-	-	-	-	-
1500 - 1599	1,259,824	1,402,474	2,348,863	3,895,270	4,698,047	4,698,106	-	-	-	-	-
1600 - 1699	627,583	780,364	1,415,147	2,257,063	3,514,829	4,698,065	4,698,092	-	-	-	-
1700 - 1799	419,889	408,444	891,818	1,499,773	2,228,952	3,999,818	4,698,035	4,698,088	-	-	-
1800 - 1899	281,902	286,769	447,341	893,978	1,468,987	2,439,832	3,665,572	4,698,000	4,698,059	-	-
1900 - 1999	190,385	171,133	272,258	448,351	818,918	1,452,505	1,940,287	3,486,275	4,697,958	4,697,311	-
>2000	228,477	186,001	277,431	423,692	651,232	1,206,468	1,718,683	2,587,573	3,967,891	4,697,533	4,697,178

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the main contributions and scope of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We discuss the limitations of our work in Section 5.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: We do not include theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We include details of model design and model training in Section 3, experimental setup in Section 4, as well as hyperparameter settings and implementation details in Appendix B.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We include the link to our code including the data processing pipeline in the abstract.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We include dataset details and hyperparameter settings in Section 4 and Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We train Maia-2 with a huge amount (9.1B) of chess positions. Therefore, it is hard to evaluate Maia-2 multiple times with different train/test splits.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the required computational resources in Appendix B.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We conduct our work under the guidance of NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the potential social impacts of our work in Section 1 and Section 5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

Justification: We discuss the limited risks of our work as a human-like model in Section 5.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We include all assets used in our paper in main texts and references.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release our code with documentations and comments.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our work does not involve crowdsourcing.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our work does not involve crowdsourcing.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.