Causal Masking on Spatial Data: An Information-Theoretic Case for Learning Spatial Datasets with Unimodal Language Models

Jared Junkin[*]
Department of Electrical and Computer Engineering
Johns Hopkins University
Baltimore, Maryland, United States
jjunkin2@jh.edu

Samuel Nathanson
Whiting School of Engineering
Johns Hopkins University
Baltimore, Maryland, United States
samuel.nathanson@jhu.edu

Abstract

Language models are traditionally designed around causal masking. In domains with spatial or relational structure, causal masking is often viewed as inappropriate, and sequential linearizations are instead used. Yet the question of whether it is viable to accept the information loss introduced by causal masking on nonsequential data has received little direct study, in part because few domains offer both spatial and sequential representations of the same dataset. In this work, we investigate this issue in the domain of chess, which naturally supports both representations. We train language models with bidirectional and causal self-attention mechanisms on both spatial (board-based) and sequential (move-based) data. Our results show that models trained on spatial board states - *even with causal masking* - consistently achieve stronger playing strength than models trained on sequential data. While our experiments are conducted on chess, our results are methodological and may have broader implications: applying causal masking to spatial data is a viable procedure for training unimodal LLMs on spatial data, and in some domains is even preferable to sequentialization.

1 Introduction

Causal masking enforces left-to-right next-token prediction and reflects the inherently sequential structure of natural language. In domains with spatial or relational data it is desirable to utilize other forms of attention which do not apply causal masking.

Natively multimodal models employ a variety of techniques to preserve causal masking across sequential data while allowing for the application of methods such as bidirectional attention, cross-attention bridges, and fused attention to non-sequential modalities. However, natively multimodal models are substantially more complicated and expensive to train due to the heavy compute and memory overhead of cross-modal attention, the difficulty of curating and balancing large paired datasets across modalities, and the burden of building modality-specific tokenization and embedding methods. This means it is desirable to explore means by which training can be made simpler or avoided altogether. This raises the question of whether it is possible to accept the information loss resulting from applying causal masking to spatial data, training a unimodal autoregressive LLM as if it were ingesting sequential data.

This question has largely gone unaddressed, due to both the existence of natively multimodal models and the fact that there are few domains where equivalent spatial and sequential representations of data

---

[*]Equal contribution.

FEN: [r1bqk2r/
1pppbppp/
p1n2n2/
4p3/
B3P3/
5N2/
PPPP1PPP/
RNBQ1RK1 w kq - 0 1]
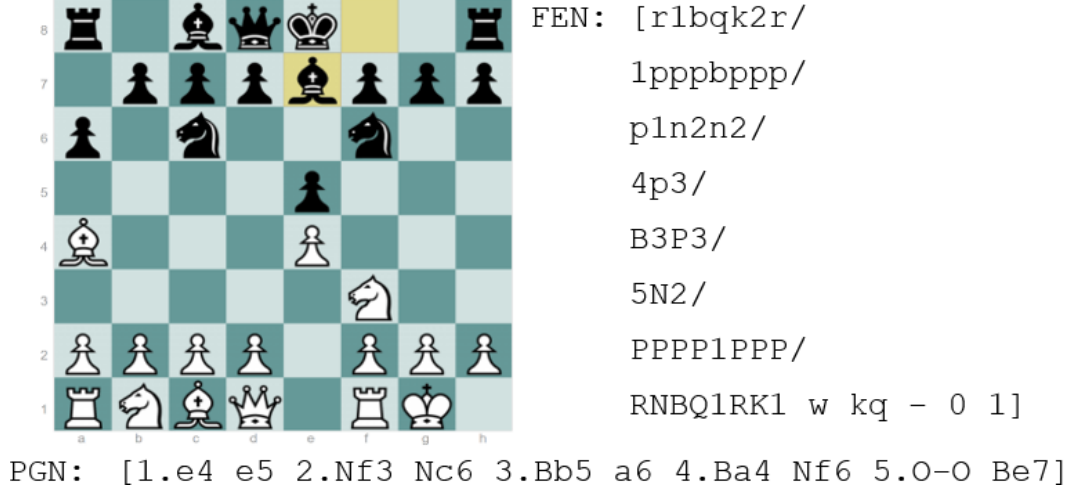
PGN: [1.e4 e5 2.Nf3 Nc6 3.Bb5 a6 4.Ba4 Nf6 5.O-O Be7]

Figure 1: PGN vs. FEN encodings for a Rúy Lopez position.

exist – a necessary prerequisite for foundational research to occur, due to the need for benchmarking and quantitative analysis of results. Chess is an ideal domain in which to examine this question. We believe the recent emphasis on smaller, cheaper language models which sacrifice exhaustive knowledge of the world in favor of greater reasoning abilities further motivates this as a timely area of research.

1.1 Learning a Transformer-Based Chess Agent

Chess offers two textual encodings: PGN (sequential moves) and FEN (spatial board state with side-to-move/rights/counters; see Appendix E). We use both; Figure 2 illustrates them for a Rúy Lopez position.

1.1.1 Learning from PGN Data

PGN data is a sequential (causal) list of moves made over the course of a chess game starting with move m_0 and ending with move m_n for a game lasting n moves. Because of this, PGN can be interpreted the same way as natural language, and a causal autoregressive LLM Π_θ can be learned on PGN that maximizes the log probability of correctly predicting the ground truth next move m_t at each timestep $t \in T$ given the sequence of preceding moves made $(m_0, m_1, \ldots, m_{t-1})$ for each sample s in some PGN Dataset $\mathcal{D}$:

$$\hat{m}_t = \arg \max_{m \in \mathcal{M}} \left[\log P(\hat{m}_t == m_t \mid m_0, m_1, \ldots, m_{t-1}) \right]$$

A categorical cross-entropy loss is imposed during training, defined as:

$$\mathcal{L} = - \sum_{i=1}^{T} \sum_{j=1}^{M} \mathbb{I}(y_{ij} == m_i) \log(y_{ij})$$

Where T gives the number of moves in a given game, $\mathbb{I}$ gives the indicator function, and M represents the size of the vocabulary space under the the given tokenization strategy. As $\mathcal{L} \to 0$, the agent's behavior approximates that of the player(s) who generated the dataset. Formally, if we assume our dataset $\mathcal{D}$ was generated by some oracle Υ which decides which move m_t to play at timestep $t \in T$ given a PGN string s representing moves $m_0, m_1, \ldots, m_{t-1}$, then for a well-trained model Π_θ:

$$\Pi_\theta(s) \approx \Upsilon(s)$$

It's worth noting that this means the model's skill is upper-bounded by the skill of the oracle Υ.

1.1.2 Learning from FEN Data

FEN is non-sequential; we train a classifier over legal moves with cross-entropy on the best move m^* (full objective in Appendix A.3).

2 Related Work

Transformer-based chess agents have followed two main paths. The first fine-tunes pretrained LLMs on chess formats: GPT-2 on PGN Noever et al. [2020], Radford et al. [2019], Biderman et al. [2023], anecdotal GPT-3.5 evaluations Acher [2023], and LLaMA-style models fine-tuned on FEN to reach professional strength Zhang et al. [2025]. These show feasibility but remain relatively rare compared to bespoke models.

A second line trains small decoder-only transformers from scratch. Karvonen Karvonen [2024] (PGN) and OthelloGPT Hazineh et al. [2023] indicate that autoregressive transformers learn latent spatial encodings from sequential moves. Ruoss et al. introduce ChessBench and strong FEN-based models with engine annotations Ruoss et al. [2024b,a]. Larger-scale efforts combine strong policies with search or alternative objectives across board games Schultz et al. [2025], Hamara et al. [2025], Ye et al. [2025]. In parallel, interpretability work shows structured world-model representations and belief-state geometry in transformers trained on symbolic/spatial tasks Toshniwal [2022], Ivanitskiy et al. [2023], Spies et al. [2024], Shai et al. [2024].

Masking strategies in spatial or multimodal settings often relax pure causal masking (e.g., block-causal for images; relaxed masks in V+L) Amrani et al. [2025], Pei et al. [2025]. By contrast, our contribution is methodological: we directly compare *causal-masked* training on spatial FEN to the conventional sequentialization path (PGN), and we show that accepting the information loss from causal masking on spatial inputs is preferable to linearizing spatial data for causal models. Additional survey details and a comparison table are provided in Appendix A.

3 Methods

3.1 Formalization of Problem

To begin with, we offer a brief formal justification for our hypothesis that training on spatial chess data is more desirable than training on sequential chess data. A model Π_θ^P trained on PGN will learn to approximate the ground truth optimal policy:

$$\Pi_\theta^P(m_0, m_1 \ldots m_{t-1}) \approx \Pi^*(m_0, m_1 \ldots m_{t-1})$$

Meanwhile, a model Π_θ^F trained on FEN will learn to approximate a first-order Markovian policy Π_{Markov} in which the best next move m^* is determined exclusively by the current board state B_t:

$$\Pi_\theta^F \approx \Pi_{\text{Markov}}(B_t) = \arg \max_{m \in M} \left[Q^*(m|B_t) \right]$$

Because chess is played with perfect information and zero-stochasticity, it is first-order Markovian with the exception of the draw by threefold repetition rule, which states that if the same board position is repeated three times, the game ends in a draw. From a mathematical perspective, if this rule did not exist, chess would obey the Markov Property, which states that the future is conditionally independent of the past given the present:

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots s_0, a_0) = P(s_{t+1}|s_t, a_t)$$

However, because of the draw by threefold repetition rule, the true optimal policy of chess is dependent on the entire sequence of board states $(B_0, B_1, \ldots B_{t-1})$ so that Π^* can avoid draws by threefold repetition in games where a winning sequence of moves exists:

$$\Pi^*(B_0 \ldots B_t) = \arg \max_{m \in M} \left[Q^*(m_i|(B_0 \ldots B_t)] \right]$$

It's also possible to express an optimal policy for determining the optimal move m^* at timestep t in terms of the sequence of moves previously made $(m_0, m_1 \ldots m_{t-1})$, because the sequence of board states can be extracted without loss of information from the sequence of moves:

$$\Pi^*(m_0, m_1 \ldots m_{t-1}) = \arg \max_{m \in M} \left[Q^*(m_i|(m_0, m_1 \ldots m_{t-1})] \right]$$

Because FEN data doesn't contain any information on past board states, any model Π_θ learned with FEN data will learn to approximate Π_{Markov}:

$$\Pi_\theta(B_t) \approx \Pi_{\text{Markov}} \neq \Pi^*(m_0, m_1 \ldots m_{t-1})$$

For a discussion of how this impacts model behavior and our solution, please see the appendix. Despite the fact that a model Π_θ^F trained on FEN will learn to approximate a suboptimal policy, we found that models trained on FEN data exhibited superior performance in practice.

Let $\mathcal{P}$ denote the space of all possible PGN strings (i.e., all sequences of valid moves from the start of a standard chess game) and $\mathcal{F}$ denote the set of all possible FEN strings. We treat $\mathcal{P}$ and $\mathcal{F}$ as countably infinite sample spaces from which we draw samples $p \in \mathcal{P}$ and $f \in \mathcal{F}$ during training. Our models Π_θ^P and Π_θ^F, models parameterized by θ and trained on PGN and FEN data respectively, learn mappings from $\mathcal{P}$ and $\mathcal{F}$ to the finite set $\mathcal{S}$, which represents all possible SAN (Standard Algebraic Notation) moves:

$$\Pi_\theta^P : \mathcal{P} \to \mathcal{S}$$

$$\Pi_\theta^F : \mathcal{F} \to \mathcal{S}$$

For a given PGN string $p \in \mathcal{P}$ there exists a surjective (many-to-one) mapping function $\mathcal{G}$ which will convert $p \in \mathcal{P}$ into a unique FEN string $f \in \mathcal{F}$:

$$\mathcal{G} : \mathcal{P} \to \mathcal{F}$$

$$\mathcal{G}(p_i) = f_i$$

The relationship between pieces on a chessboard (which pieces threaten which, which pieces guard which) is determined by their spatial orientation to each other. For example, bishops threaten pieces along their active diagonal; rooks threaten pieces along the file or row. Because of this, a policy for choosing the best move Π^* must condition on *spatial* information. While the *data* in PGN is structured sequentially, the ground truth information the model must reason against when evaluating move choices is *structured spatially*. Therefore a model Π_θ^P must back out some latent spatial representation of the board state B from the PGN sequence $p \in \mathcal{P}$ in order to generate the best move $s \in \mathcal{S}$; that is, a model Π_θ^P trained on PGN data actually learns a composition of functions:

$$\Pi_\theta^P : \mathcal{G} \circ (\mathcal{F} \to \mathcal{S})$$

Where $\mathcal{G}$ is the surjective mapping of PGN data into the FEN space (or, more specifically, into some latent dimension approximating a spatial reconstruction of board state B), while a model Π_θ^F trained from FEN data does not have to perform this composition, as the information is already structured spatially:

$$\Pi_\theta^F : \mathcal{F} \to \mathcal{S}$$

We hypothesize that the composition $\mathcal{G} \circ (\mathcal{F} \to \mathcal{S})$ entails greater representational complexity than the direct mapping from FEN to SAN moves $\mathcal{F} \to \mathcal{S}$. Intuitively, PGN-based models must internally reconstruct spatial board states before selecting moves, whereas FEN-based models can condition directly on spatial structure. We present this as an information-theoretic intuition rather than a formal proof.

3.2 Models & Datasets

We conduct our experiments on Meta AI's 1.3B Parameter Llama3.1 Model Llama Team [2024] and two identically-sized character-level language models. One character-level language model was trained on PGN data with causal masking, and the other was trained on FEN data without causal masking; meanwhile the Llama model was trained using FEN data *while using causal masking*. For our FEN dataset, we utilized the Chessbench dataset provided by Ruoss et al. [2024b], which consists of 15 billion board positions annotated with the top move according to Stockfish Stockfish Developers [2024], which is the world's strongest chess engine as of January 2025. Our PGN dataset consisted of approximately 1 Billion PGN strings representing games played between a maximum-strength Stockfish agent as white and a variety of attenuated chess engines with ELO ratings between 1200 (advanced beginner) and 3100 (superhuman skill) playing as black. The dataset was assembled this way to ensure that neither moves played by beginners nor experts were out-of-distribution. Our PGN model only played as white in the simulated tournaments we utilized for evaluation, while both the models trained on FEN played both white and black.

3.3 Prompting & Tokenization

For FEN, we enforce character-level tokenization to avoid ambiguous merges and align with the pretrained vocabulary; prompts embed the FEN, the legal SAN moves, and the engine best move, which stabilizes training. Details on merges/run-length handling, templates, and padding appear in Appendix F.

Full tokenization templates, padding details, and examples are provided in Appendix F.

3.4 Objective Function

Let X be the tokenized prompt described above. Let the tokens in this prompt representing the best move be denoted by by $m^* = (m_1^*, \ldots, m_k^*)$, and let the tokens we predict for the best move be denoted as $\hat{m} = (\hat{m}_1, \hat{m}_2 \ldots \hat{m}_k)$. We create a binary loss mask $\mathbf{w} = (w_1, \ldots, w_T)$ of length $T = |X|$, where

$$w_t \;=\; \begin{cases} 1, & \text{if token } t \text{ is part of } m^*, \\ 0, & \text{otherwise.} \end{cases}$$

Let the model's predicted distribution at step t be

$$p_\theta\big(y_t \mid X\big),$$

where θ are the model parameters. Our objective function is a masked cross-entropy loss wherein we only sum over tokens belonging to m^*. All other predictions are masked out (where $w_t = 0$), contributing no penalty to $\mathcal{L}$:

$$\mathcal{L}_{\text{masked}}(\theta) = -\sum_{t=1}^{T} w_t \log p_\theta\Big(m_t^* \mid X_{0:t-1}\Big),$$

We use a typical lower-triangular causal attention mask, so that at step t, the model can only attend to tokens up to $t - 1$. Padding tokens are also masked. It's worth emphasizing that because the ground truth target tokens $(m_{<t}^* = m_0^* \ldots m_{t-1}^*)$ are also a part of our tokenized prompt $X_{0:t-1}$, our objective function is fully *teacher forced* Williams and Zipser [1989]. The final gradient update to θ is thus driven solely by errors in predicting the best-move tokens, and each target token m_i^* conditions on the full prompt (including the target tokens preceding it).

Exposure bias (brief). We observed only modest differences between teacher-forced and autoregressive decoding; full metrics and figures are provided in Appendix C.

4 Experiments

4.1 Training Details

All models trained for 200,000 steps on $2 \times$A100 (80GB) with cosine LR decay, warmup, gradient clipping, and mixed precision. We followed published hyperparameters for Llama and Karvonen-style settings for the character baselines. Results are from a single training run per game due to compute constraints (each run 3 weeks on our hardware). Full configurations and loss curves are reported in Appendix D and Appendix G.

4.2 Empirical Validation that Training on Spatial Data Produces Superior Skill

Our first experiment is an empirical justification of the hypothesis formulated in the previous section. As mentioned in the introduction, we trained two identical NanoGPT models on equivalently sized datasets of PGN and FEN data. The only difference between these two models is that the model trained on PGN utilized causal self-attention with lower-triangular masking, while the model trained on FEN utilized bidirectional attention with no masking, allowing it to attend to all tokens simultaneously because FEN data is spatial. We trained both models for 200,000 training steps with equal batch sizes and hyperparameter values. We then evaluated the mean cross-entropy per sample for FEN and PGN test data:

(a) Cross Entropy Loss
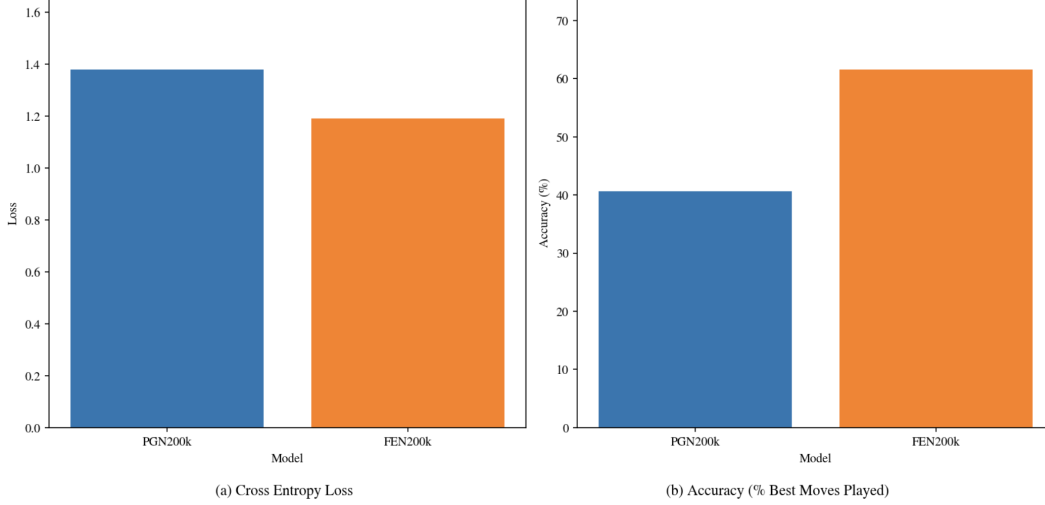
(b) Accuracy (% Best Moves Played)

Figure 2: The total cross entropy loss per prediction (left) and accuracy in % of top engine moves played (right) for models trained on FEN (blue) and PGN (yellow)

$$\text{CE}_{\text{PGN}} = -\frac{1}{N} \sum_i \sum_{t \in |p_i|} \log P_{M_\theta^P}(\hat{m}_{it}|p_i)$$

$$\text{CE}_{\text{FEN}} = -\frac{1}{N} \sum_i \log P_{M_\theta^F}(\hat{m}_i|f_i)$$

Where $\hat{m}_i$ gives the model's predicted move for the ith sample. The equation for PGN contains a second sum because a single move will contain t tokens; in FEN data, the tokenization scheme represents each move as a single token.

We observed that the model trained on PGN did indeed have substantially higher loss values and lower best-move classification when compared to the model trained on FEN.

4.3 Causal Masking on Spatial Data

Next we applied the same training regimen to our causal Llama model, training it to play chess via a large-scale supervised fine tuning run on our FEN dataset. The model displays grandmaster-level performance with an estimated ELO rating of 2630, approximately 500 points higher than that of the causal model trained on PGN and only narrowly worse than the bidirectional model trained on FEN. These results demonstrate that when training causal language models to play chess, it is better to accept the information loss from applying causal masking to sequential FEN data than to linearize the data into a sequential representation beforehand.

On a held-out set of $\sim 12,800$ positions (zero-shot baseline compared to performance after SFT):

- **Syntactically valid move rate:** 99.9% after SFT.
- **Legal move rate:** $\approx 100\%$ after SFT.
- **Best move (Stockfish):** $\sim 0.6\% \rightarrow \approx 58\%$ ($\sim 100\times$ increase).

We then compared this to both of our other models' inference-time performance on the same three metrics (there was no sense baselining either of the other two models, as we trained them from scratch rather than conducting SFT). All three models are capable of outputting valid and legal moves with nearly 100% accuracy; however, there is a clear gradient in their abilities to output the best Stockfish move. The model trained on PGN outputs the best move only 40% of the time, while our Llama model outputs the best move 58% of the time and the bidirectional model trained on FEN outputs the best move over 60% of the time.
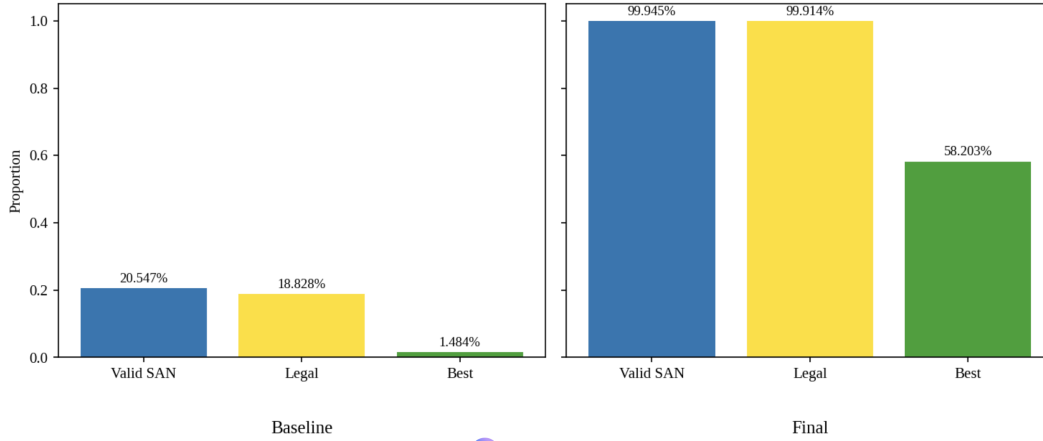
6

Figure 3: The % of prompts resulting in Syntactically Valid, Legal, and Best (according to chess engine) moves. *Note:* All numbers are from a single run per game. See Appendix D for hardware/runtime.
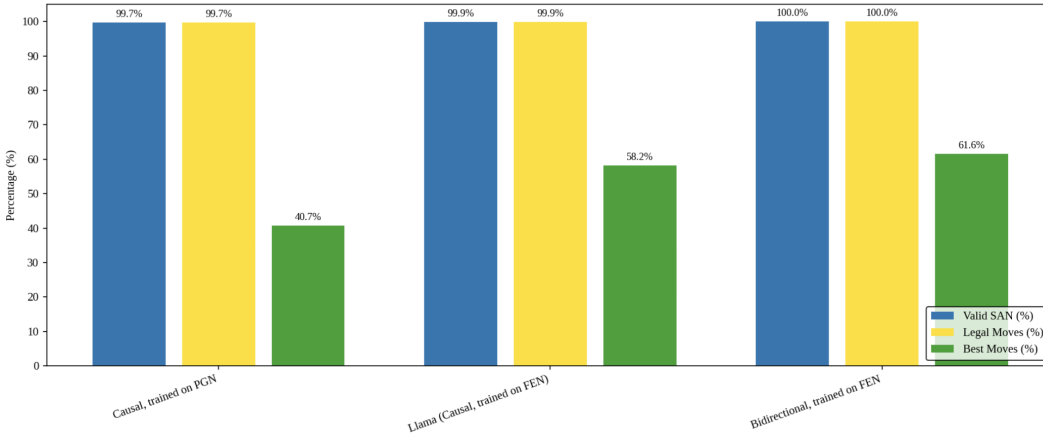

Figure 4: The Percentage of Prompts resulting in Syntactically Valid, Legal, and Best (according to chess engine) Moves.

4.4 Calculating Elo Rating

We estimated Elo by playing each model against calibrated Stockfish agents (Levels 0–10), using 1000 games per level (balanced White/Black) and standard Elo updates. We report headline Elo in the main text; full protocol, equations, and the Stockfish calibration table appear in Appendix I and Appendix H.

5 Discussion

This paper presents a case study of a causal-masked LLM that demonstrates grandmaster-range performance under Elo evaluation against Stockfish agents and demonstrates that, when training a casual language model to play chess, it is better to accept the information loss that results from applying causal masking to spatial data than to linearize board state into a sequential representation ahead of causal masking. We have presented a compelling theoretical case that this is due to the higher functional complexity resulting from having to back out latent representations of relevant spatial features in the board state, and backed it up with an empirical analysis. With a high-quality dataset,

sophisticated prompting and tokenization strategies, and adherence to published hyperparameter values and best practices training principles, it is possible to learn highly sophisticated domain-specific behavior from non-causal data with causal LLMs – even with small LLMs and modest compute resources. We believe that abstract strategy games such as chess remain incredibly fertile ground for AI research, and hope this result stimulates future research across other domains in machine learning. To that end, we conclude our paper with several avenues we think warrant further exploration, as well as key lessons learned which we hope will aid future researchers:

5.1   Scaling Laws, Pretraining Quality, & Overparameterization

We also experimented with training a 140M parameter Pythia model to play chess. We observed Llama consistently outperformed Pythia even when the two models were trained on equivalent batch sizes for identical numbers of training steps.

However, both models are *overparameterized* for the task of learning chess, because Ruoss Ruoss et al. [2024b,a] and Karvonen Karvonen [2024] both published results with 50M parameter models that reached high skill levels in chess. This means that either *A)* the quality and duration of pretraining is a significant determinant in model performance, or *B)* the number of non-embedding parameters in a LLM is a significant determinant in performance *even when* both models are overparameterized relative to the problem. Exploring which of these, or both, is the case, would be an interesting avenue to for future resesarch.

5.2   Chess & LLM Interpretability

Abstract strategy games like chess have a far smaller vocabulary size than natural language, intuitive visual interpretations, and quantitative means of assessing response quality and correctness. Because of this, it's an extremely interesting sandbox for evaluating and developing new methods for AI interpretability.

5.3   Importance of Tokenization Process

In our experiments, we found that tokenization strategy had a significant effect on training convergence. Both the Pythia and Llama models failed to converge under their default tokenizers, which merge character sequences such as "pk" (pawn–king) into a single token. To address this, we modified the tokenizer to operate strictly at the character level and flattened run-length encodings in FEN strings. These changes ensured that each chess piece and empty square was represented consistently, and that counters were aligned across all samples. After this modification, models trained stably and achieved strong performance. We interpret this result as evidence that a well-matched tokenization schema is an important factor when adapting pretrained models to highly structured symbolic domains such as chess. While we cannot rule out that alternative strategies might also succeed, our findings highlight that careful preprocessing is often necessary to align domain-specific inputs with the statistical assumptions of pretrained tokenizers.

6   Limitations

Our work has several limitations. First, Elo ratings were computed against attenuated Stockfish agents; while these provide a calibrated baseline, they are not directly equivalent to FIDE Elo ratings for human players. Second, models trained on FEN data are blind to the threefold repetition rule, meaning they cannot avoid certain draws without additional mechanisms. Third, our results depend on a single training run per condition with modest compute, limiting variance estimates. Finally, while we hypothesize broader implications of applying causal masking to spatial domains, our empirical results are confined to chess; generalization to other domains remains future work.

7   Conclusion

Chess has been an active area of AI research for so long that the game is commonly referred to as "the Drosophila of artificial intelligence." In this work, we introduced, to our knowledge, the first open-source fine-tuned LLM to achieve performance in the ∼2600 Elo range when calibrated against

Stockfish agents. More importantly, we provided a methodological case study showing that causal masking on spatial data, while lossy, can outperform sequentialization, and that careful tokenization and prompting strategies allow pretrained LLMs to exploit structured symbolic domains effectively.

Our first key insight is that, because the relationships between pieces on a chess board are spatial, any model

$$\Pi_\theta^P : \mathcal{P} \to \mathcal{S}$$

trained to output chess moves from sequential PGN inputs must still recover spatial board-state information. While this information exists in compressed form within PGN and can be extracted losslessly, the model must devote additional capacity to forming a latent spatial representation

$$\mathcal{G} \circ (\mathcal{F} \to \mathcal{S}),$$

which approximates this reconstruction. Intuitively, this mapping entails greater representational complexity than training a model

$$\Pi_\theta^F : \mathcal{F} \to \mathcal{S}$$

directly on spatial FEN data. We hypothesize that this explains why, in practice, FEN-trained models tend to achieve stronger performance, even though they operate under causal masking and lose some bidirectional information.

Our second key insight was the importance of aligning tokenization and prompting with the symbolic structure of the domain. By enforcing character-level tokenization for FEN and embedding domain-specific information directly into prompts, we ensured stable training and enabled LLMs to represent board states consistently. This highlights how tokenizer alignment is not a technical afterthought, but a central design choice when adapting pretrained models to structured domains.

Our findings highlight chess as a fertile and interpretable sandbox for studying representation learning, masking strategies, and world-model formation in LLMs. While our results are confined to chess, the methodological lessons—particularly around spatial causal masking and tokenizer alignment—may extend to other spatially structured domains. For the LAW community, we view this as evidence that causal masking can remain a viable design choice for structured reasoning tasks, and that abstract games provide a uniquely transparent setting in which to probe questions about world models and representation.

We hope that this study not only contributes a strong open-source baseline for chess-playing LLMs, but also motivates further exploration of causal masking and representation learning in broader spatial reasoning contexts.

References

Mathieu Acher. Debunking the chessboard: Confronting gpts against chess engines to estimate elo ratings and assess legal move abilities. Mathieu Acher Blog, 2023. URL `https://blog.mathieuacher.com/GPTsChessEloRatingLegalMoves/`.

Gil Amrani, Yuval Alaluf, and Tomer Michaeli. Sample- and parameter-efficient auto-regressive image models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. URL `https://arxiv.org/abs/2411.15648`.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint*, 2023. URL `https://arxiv.org/pdf/2304.01373`.

Computer Chess Rating Lists. Top chess engines. CCRL Website, 2025. URL `https://computerchess.org.uk/ccrl/4040/`.

FIDE. Top chess players. FIDE Ratings Website, 2025. URL `https://ratings.fide.com/`.

Felix Hamara, Aaron Smith, and Rachel Lee. Learning to plan via supervised contrastive learning and strategic interpolation: A chess case study. *arXiv preprint*, 2025. URL `https://arxiv.org/abs/2506.04892`.

Loay Hazineh, Tianyu Li, and Zhen Wang. Linear latent world models in simple transformers: A case study on othello-gpt. *arXiv preprint*, 2023. URL `https://arxiv.org/abs/2310.07582`.

Grigory Ivanitskiy, Nikita Pavlichenko, and Dmitry Petrov. Structured world representations in maze-solving transformers. *arXiv preprint*, 2023. URL `https://arxiv.org/abs/2312.02566`.

Adam Karvonen. Emergent world models and latent variable estimation in chess-playing language models. *arXiv preprint*, 2024. URL `https://doi.org/10.48550/arXiv.2403.15498`. Accepted to the 2024 Conference on Language Modeling.

AI @ Meta Llama Team. The llama 3 herd of models. *arXiv preprint*, 2024. URL `https://arxiv.org/pdf/2407.21783`.

David A. Noever, Matt Ciolino, and Josh Kain. The chess transformer: Mastering play using generative language models. *arXiv preprint*, August 2020. URL `https://arxiv.org/pdf/2008.04057`.

Yifan Pei, Junjie Li, and Tao Wang. Rethinking causal mask attention for vision-language inference. *arXiv preprint*, 2025. URL `https://arxiv.org/abs/2505.18605`.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI*, 2019. URL `https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`.

Anian Ruoss, Grégoire Delétang, Sourabh Medapati, Jordi Grau-Moya, Kevin Li Wenliang, Elliot Catt, John Reid, Cannada Lewis, Joel Veness, and Tim Genewein. Grandmaster-level chess without search. *arXiv preprint*, 2024a. URL `https://arxiv.org/abs/2402.04494`.

Anian Ruoss, Grégoire Delétang, Sourabh Medapati, Jordi Grau-Moya, Li Kevin Wenliang, Elliot Catt, John Reid, Cannada A. Lewis, Joel Veness, and Tim Genewein. Amortized planning with large-scale transformers: A case study on chess. *arXiv preprint*, 2024b. URL `https://doi.org/10.48550/arXiv.2402.04494`.

Laura Schultz, Hao Chang, and Yilun Du. Mastering board games by external and internal planning with language models. *arXiv preprint*, 2025. URL `https://arxiv.org/abs/2412.12119`.

Daniel Shai, Youngsoo Kim, and Mei Xu. Transformers represent belief state geometry in their residual stream. *arXiv preprint*, 2024. URL `https://arxiv.org/abs/2405.15943`.

Lukas Spies, Chelsea Chang, and Jack Clark. Transformers use causal world models in maze-solving tasks. *arXiv preprint*, 2024. URL `https://arxiv.org/abs/2412.11867`.

Stockfish Community Stockfish Developers. Stockfish: A free and strong uci chess engine. GitHub repository, 2024. URL `https://github.com/official-stockfish/Stockfish`.

Shubham Toshniwal. Chess as a testbed for language model state tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 2022. URL `https://ojs.aaai.org/index.php/AAAI/article/view/21390`.

R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989. doi: 10.1162/neco.1989.1.2.270. URL `https://doi.org/10.1162/neco.1989.1.2.270`.

Mingxuan Ye, Jaewon Park, and Kai Zhou. Implicit search via discrete diffusion: A study on chess. *arXiv preprint*, 2025. URL `https://arxiv.org/abs/2502.19805`. ICLR 2025.

Wei Zhang, Li Chen, and Arjun Kumar. Complete chess games enable llm become a chess master. *arXiv preprint*, 2025. URL `https://arxiv.org/abs/2501.17186`.

A   Expanded Related Work

A.1   Narrative overview

We group prior work into five themes. (i) **Fine-tuning pretrained LLMs on chess** shows feasibility but is relatively sparse: GPT-2 on PGN Noever et al. [2020], Radford et al. [2019], Biderman et al. [2023], anecdotal GPT-3.5 strength Acher [2023], and LLaMA-style models fine-tuned on FEN to professional levels Zhang et al. [2025]. (ii) **Small models from scratch** demonstrate that even nano-scale GPTs trained on PGN or FEN can learn latent spatial structure and achieve strong play Karvonen [2024], Hazineh et al. [2023], Ruoss et al. [2024b]. (iii) **Scaling to grandmaster-range** uses larger models, engine annotations, and/or search to reach superhuman strength Ruoss et al. [2024a], Schultz et al. [2025], Hamara et al. [2025], Ye et al. [2025]. (iv) **Interpretability/world models** finds that transformers trained on symbolic/spatial domains encode legal-state tracking, causal world models, and belief-state geometry Toshniwal [2022], Ivanitskiy et al. [2023], Spies et al. [2024], Shai et al. [2024]. (v) **Masking strategies** adapt or relax causality for spatial inputs Amrani et al. [2025], Pei et al. [2025].

Our work differs by *explicitly* comparing (a) PGN sequentialization under causal masking and (b) *spatial* FEN under causal masking, showing the latter outperforms despite information loss from masking.

Table 1: Representative prior work across five themes. "Masking" indicates the dominant attention style in the cited setup.

Theme	Representative works	Representation	Masking	Key takeaway
Fine-tuning LLMs	GPT-2 on PGN ciolino,gpt2,pythia2023; GPT-3.5 notes Acher [2023]; LLaMA on FEN Zhang et al. [2025]	PGN / FEN	Causal (PGN), mixed (FEN)	Feasible to adapt pre-trained LLMs; FEN fine-tuning can reach pro-level strength.
Small models from scratch	Karvonen (PGN) Karvonen [2024]; Othel-loGPT Hazineh et al. [2023]; ChessBench Ruoss et al. [2024b]	PGN vs. FEN	Causal (PGN); bidirectional (FEN)	Transformers learn latent spatial world models; FEN often stronger in practice.
Scaling to GM-range	Ruoss et al. Ruoss et al. [2024a]; Schultz et al. Schultz et al. [2025]; Hamara et al. Hamara et al. [2025]; Ye et al. Ye et al. [2025]	PGN/FEN + search or alt. objectives	Mixed	Large data/models (and search) reach superhuman strength across board games.
Interpretability / world models	Toshniwal Toshniwal [2022]; Ivanitskiy Ivanitskiy et al. [2023]; Spies Spies et al. [2024]; Shai Shai et al. [2024]	Sequential → latent spatial	Mostly causal	Structured representations (legal-state tracking, causal models, belief-state geometry) emerge.
Masking strategies (spatial inputs)	Amrani (block-causal images) Amrani et al. [2025]; Pei (relaxed V+L masks) Pei et al. [2025]	Images / V+L	Block/relaxed causal	Causality can be adapted rather than abandoned for spatial modalities.

A.2   Notes and contrasts to our study

- Prior FEN work typically *avoids* causal masking or uses bidirectional attention; we instead *accept* causal masking on spatial inputs and find it preferable to PGN sequentialization.
- Our comparison holds model/data size constant across settings where possible, isolating the masking/representation choice.

- Results align with interpretability findings that spatial structure emerges even from sequential inputs, but we show leveraging *explicit* spatial inputs with causal masking is stronger.

A.3   Objective Function

FEN notation represents the 2D spatial orientation of the pieces in a given board state and is therefore non-causal. Because of this, each board state B is instead paired with a ground-truth label m^*, representing the best next move according our oracle Υ. The objective now is to learn a model that maximizes the log probability of correctly predicting the best move m^* given board state B:

$$\hat{m} = \arg\max_{m \in \mathcal{M}} \left[\log P(m == m^* \mid B) \right]$$

With loss function:

$$\mathcal{L} = -\sum_{j=1}^{M} \mathbb{I}(y_j == m^*) \log(y_j)$$

This loss function is the same as above, except now it is *not* computed across all timesteps comprising a game, because FEN has no temporal dimension; only the target token(s) comprising $\hat{m}$ contribute to the loss.

B   Empirical Demonstration of Functional Complexity Difference between FEN and PGN

As a preliminary step ahead of our full training loop, we decided to test our theory that a mode Π_θ^F trained on FEN will exhibit higher performance than an equivalent model Π_θ^P trained on PGN due to the challenge of learning an accurate spatial representation from PGN data. To this end, we trained Karvonen's 50M parameter decoder-only NanoGPT on equivalently sized datasets of PGN and FEN data. We trained both models for 200,000 training steps with equal batch sizes and hyperparameter values. We removed the causal attention mask from the nanoGPT model to be trained on FEN, allowing it to attend to all tokens simultaneously, because FEN data is spatial. We then evaluated the mean cross-entropy per sample for FEN and PGN test data:

$$\text{CE}_{\text{PGN}} = -\frac{1}{N} \sum_i \sum_{t \in |p_i|} \log P_{M_\theta^P}(\hat{m}_{it}|p_i)$$

$$\text{CE}_{\text{FEN}} = -\frac{1}{N} \sum_i \log P_{M_\theta^F}(\hat{m}_i|f_i)$$

Where $\hat{m}_i$ gives the model's predicted move for the ith sample. The equation for PGN contains a second sum because a single move will contain t tokens; in FEN data, the tokenization scheme represents each move as a single token.

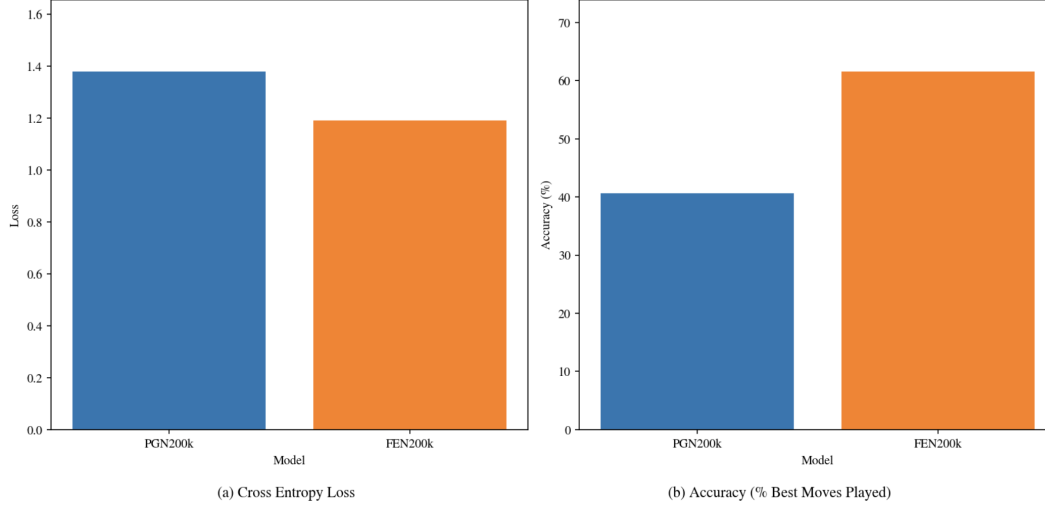| (a) Cross Entropy Loss | (b) Accuracy (% Best Moves Played) |

Figure 5: The total crossentropy loss per prediction (left) and accuracy in % of top engine moves played (right) for models trained on FEN (blue) and PGN (yellow).

We observed that the model trained on PGN did indeed have substantially higher loss values and lower best-move classification when compared to the model trained on FEN.

C Quantifying Exposure Bias in Trained Models

C.1 Formal Definition

Exposure bias refers to the mismatch between training, where models condition on ground-truth tokens, and inference, where they condition on their own generated tokens. Formally:

$$p_\theta\big(y_t \mid y_{<t}^{\text{model}}\big) \; \neq \; p_\theta\big(y_t \mid y_{<t}^{\text{ground-truth}}\big),$$

so the model encounters different distributions at train and test time.

D Training Configurations and Hyperparameters

D.1 Compute Setup

We trained all models on two Nvidia A100 GPUs with 80GB VRAM each. Training ran for 200,000 steps using a cosine decaying learning rate, 2000 warmup steps, and gradient clipping at ± 1.0. Mixed-precision training was enabled to improve memory efficiency, and gradient accumulation was used to simulate larger batch sizes.

D.2 Loss Curves

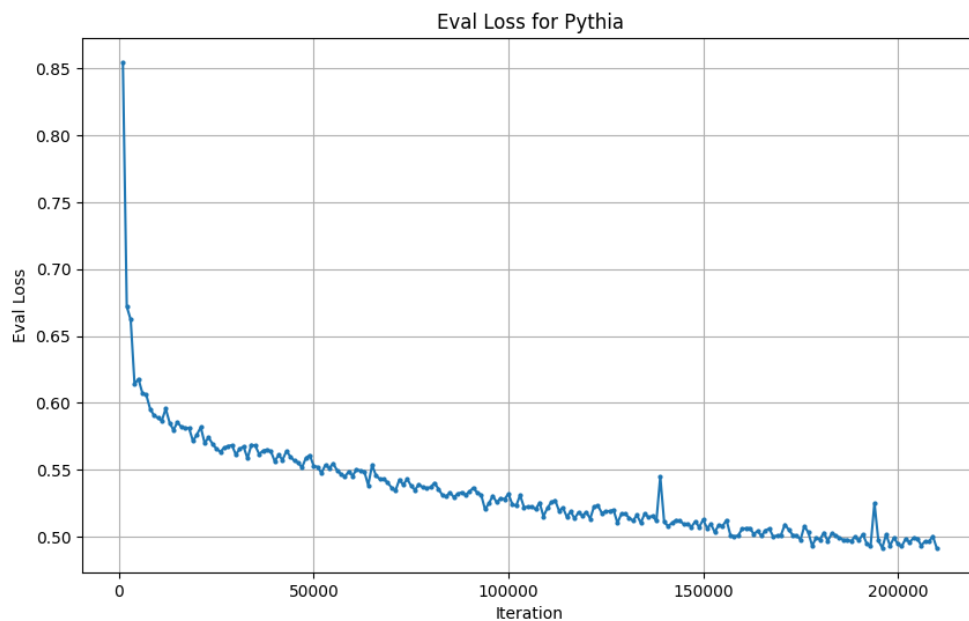Figures 6–7 show training loss curves.

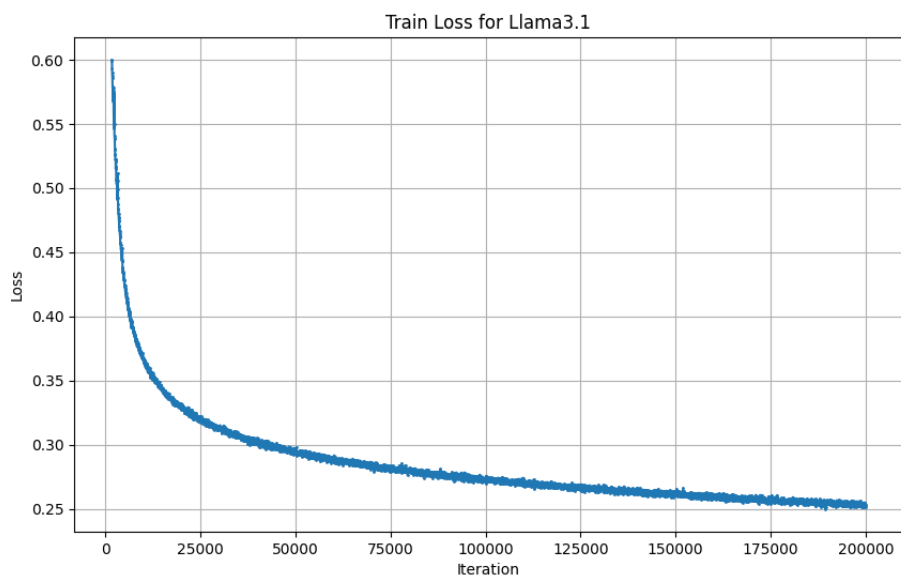Figure 6: Pythia training loss. Greater variance due to smaller batch size.

Figure 7: Llama training loss.

D.3 Hyperparameters

D.4 Training Dynamics

We observed that increasing effective batch size sped convergence sublinearly while increasing runtime linearly. This required balancing accuracy against compute budget. Despite this, all models converged to strong performance.

Table 2: training hyperparameters for Llama and Pythia models.

Hyperparameter	Llama	Pythia
Learning rate	8e-05	3e-04
Batch size	128	256
Gradient accumulation steps	32	4[*]
Token batch size	32,768	8,192
Max iterations	200,000	200,000
β_1	0.9	0.9
β_2	0.95	0.95
Warmup iterations	2,000	2,000
LR decay iterations	200,000	200,000
Minimum LR	5e-06	6e-05
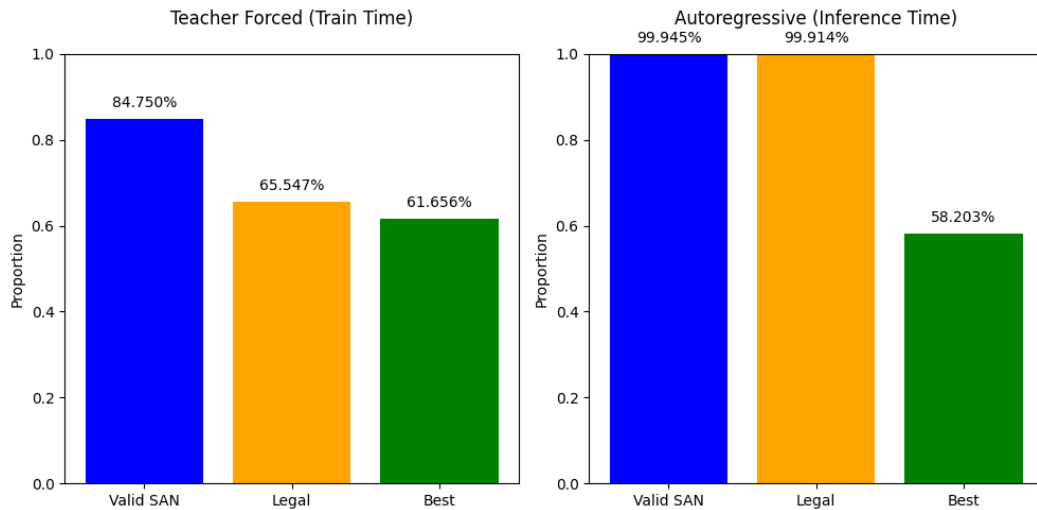
Figure 8: Llama exposure bias analysis.

D.5 Experimental Setup

We prompted our fully trained Llama and Pythia models with 12,800 board positions sampled from our test dataset. Both models generated predictions under two conditions: 1. **Teacher-forced next-token generation** (conditioning on ground-truth tokens). 2. **Autoregressive generation** (conditioning on the model's own predictions).

We then compared performance on three metrics: - (%) syntactically valid chess moves, - (%) legal moves in the given board state, - (%) best move according to Stockfish.

D.6 Results

We found that Llama and Pythia incurred modest penalties with respect to best-move prediction at inference time (Llama: $\sim 3\%$ drop; Pythia: $< 1\%$ drop). Interestingly, both models were more likely to generate legal and valid SAN moves in autoregressive mode, suggesting they internalized structural constraints of chess notation.

E Background: PGN and FEN

PGN (Portable Game Notation). A text format that records a game as a *sequential* list of moves (typically in SAN), optionally preceded by metadata (event, site, players, etc.). Applying the legal move sequence from any PGN prefix deterministically produces a unique board position.
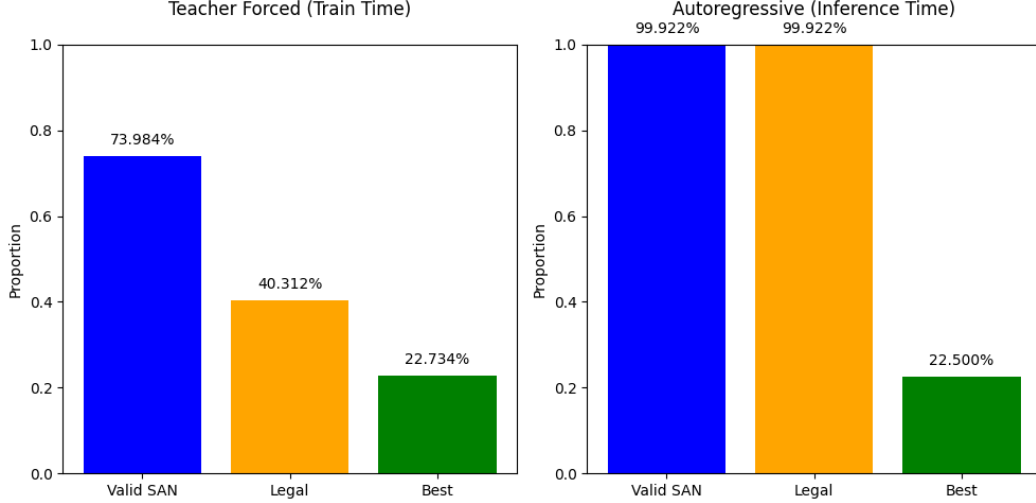
Figure 9: Pythia exposure bias analysis.

FEN (Forsyth–Edwards Notation). A *spatial* snapshot of a single position using six fields: (1) piece placement (ranks 8→1 separated by "/"; digits count consecutive empty squares), (2) active color (`w` or `b`), (3) castling availability (`KQkq` subset or "-"), (4) en passant target square or "-", (5) halfmove clock (for the 50-move rule), (6) fullmove number.

Example. `rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1` denotes the initial position: White to move; all castling rights available; no en passant target; zero halfmoves; fullmove 1.

Relation. There is a deterministic mapping $\mathcal{G} : \mathcal{P} \to \mathcal{F}$ from a legal PGN prefix $(m_0, \ldots, m_{t-1})$ to its FEN position f_t, i.e., $\mathcal{G}(p_t) = f_t$. See Figure 2 in the main text for a side-by-side illustration.

F   Detailed Tokenization and Prompting Strategy

For FEN data, our bidirectional model used the tokenization approach from Ruoss et al., while our causal Llama model used a modified version designed to better align with pretrained tokenization. In particular, we enforced character-level tokenization to avoid ambiguous merges (e.g., "pk" for pawn–king) and flattened run-length encodings. We also embedded FEN strings alongside the list of legal SAN moves and the best Stockfish move in a templated prompt. This preprocessing stabilized training and improved convergence.

Flattening FEN strings. We flattened run-length encodings in FEN to ensure consistency (e.g., `r1bqk2r/` → `r.bqk..r/`). Periods signify empty squares.

Character-level tokenization. We forced the tokenizer to tokenize strictly at the character level. For example, Llama's tokenizer would otherwise tokenize "pk" as a single token `[21486]`, but our approach tokenized it as `['p', 'k'] → [79, 74]`.

Prompt templates. Our final prompts embedded FEN strings, the list of legal SAN moves, and the Stockfish best move, in a templated instruction. For example:

```
<|begin_of_text|> You are a chess grandmaster.  This is the
board in FEN notation:  {FEN}.  The legal moves are:  {List
of Legal Moves}.  Which of these is the best move?  Best move:
{Best Move} <|end_of_text|>
```

We evaluated more than a dozen prompt variants (Appendix K), selecting the above based on zero-shot performance. We also padded sequences to handle up to 60 legal moves and promotions requiring 5 tokens, covering 99.95% of positions in the dataset.

G   Loss Curves for Training

Note that the greater variance in the loss curve for Pythia is due to a combination of smaller batch size and the fact that we switched our logging system to log the running mean loss across training steps before starting Llama.

H   Stockfish Ratings

Table 3: Stockfish levels and their approximate Elo ratings used for calibration.

Stockfish level	Elo rating
0	1320
1	1467
2	1608
3	1742
4	1922
5	2203
6	2363
7	2499
8	2596
9	2702
10	2788

I   Elo Rating Evaluation

I.1   Overview of the Elo System

Chess players' skill is typically quantified using the Elo rating system. Elo ratings are not absolute, but relative to those of other players, and are determined by comparing expected vs. actual performance. As of 2025, the world's strongest human player is Magnus Carlsen, with an Elo rating of 2831 FIDE [2025]. Stockfish, the world's strongest chess engine, has an estimated Elo rating of 3642 Computer Chess Rating Lists [2025].

I.2   Evaluation Protocol

To estimate Elo ratings, we played each model in a large-scale simulated tournament against attenuated Stockfish agents with known approximate strengths (Levels 0–10, see Table 3). Each model played: - 1000 games per Stockfish level (500 as White, 500 as Black). - The PGN-trained model played only as White, due to dataset limitations. - To ensure diverse game states, we used an opening book at the start of each game. - Moves were generated with temperature-based stochastic sampling. If more than 5 consecutive illegal moves were produced, the game was terminated and counted as a loss. (In practice, none of our models forfeited games this way.)

I.3   Elo Computation

Elo ratings were computed relative to the known ratings of the Stockfish agents. The Elo update formula was:

$$R_{\text{new}} = R_{\text{current}} + K \cdot (W - N \cdot E)$$

where W is the number of wins, N the total games, E the expected win rate, and $K = 16$. The expected win rate was defined as:

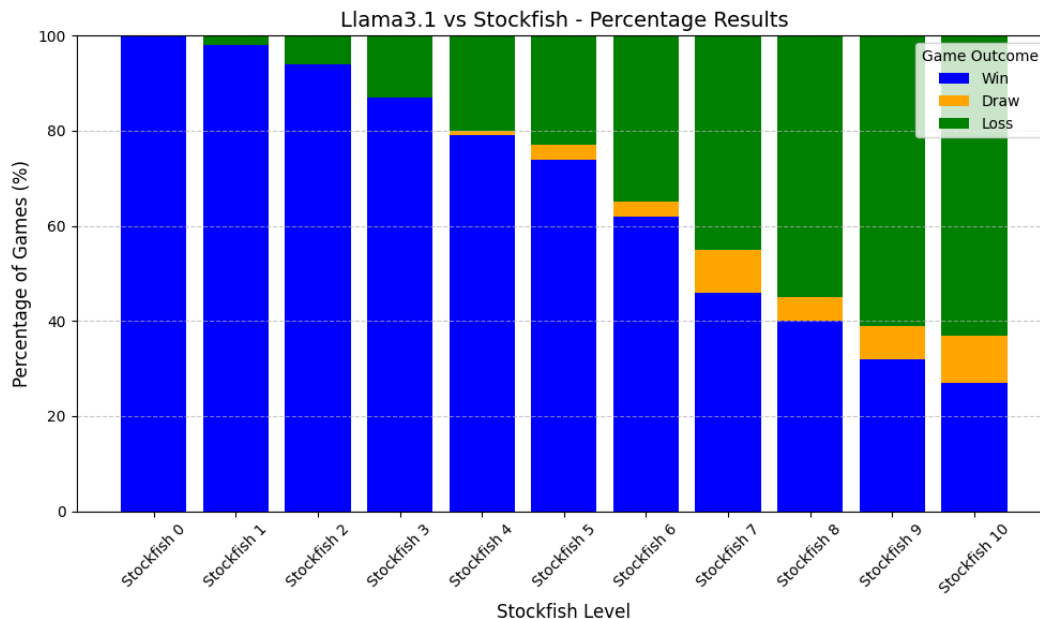$$E = \frac{1}{1 + 10^{\frac{R_{\text{opponent}} - R_{\text{current}}}{400}}}.$$

17

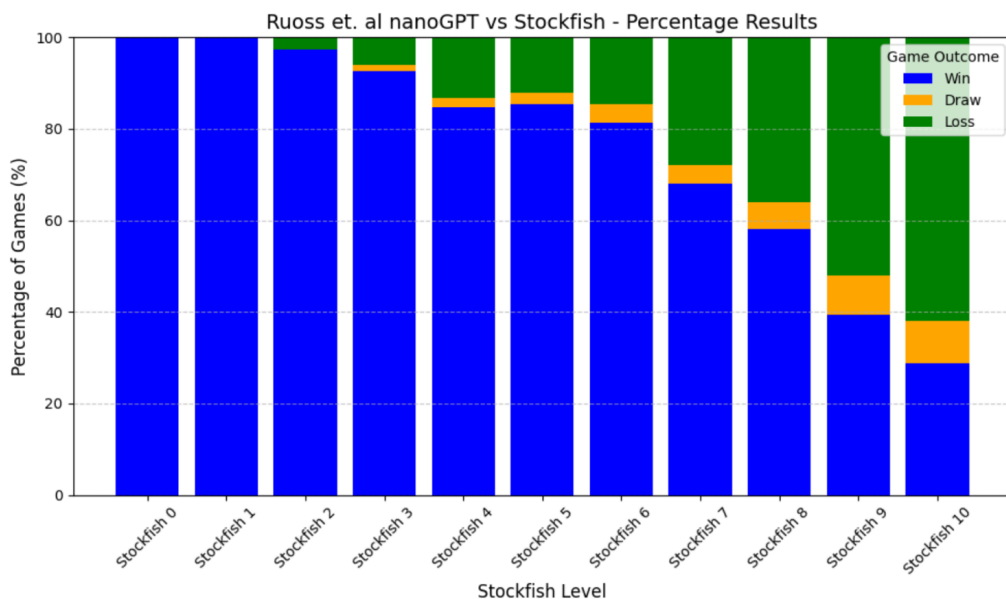Figure 10: Llama wins out of 1000 against Stockfish 0–10.

Figure 11: Ruoss et al. wins out of 1000 against Stockfish 0–10.

I.4   Results

Our Llama model shows extraordinarily strong performance, winning or drawing more than half of its games against Stockfish 7 (approx. Elo 2500, grandmaster level), and even managing results against Stockfish 10. Its estimated Elo rating based on Stockfish evaluations is 2630, stronger than approximately 90% of human grandmasters. This makes it the strongest chess-playing LLM to date, far surpassing GPT-3.5 Turbo's estimated Elo of 1750.

Our Pythia model performs relatively poorly, winning games only against the first six Stockfish agents and losing heavily overall.
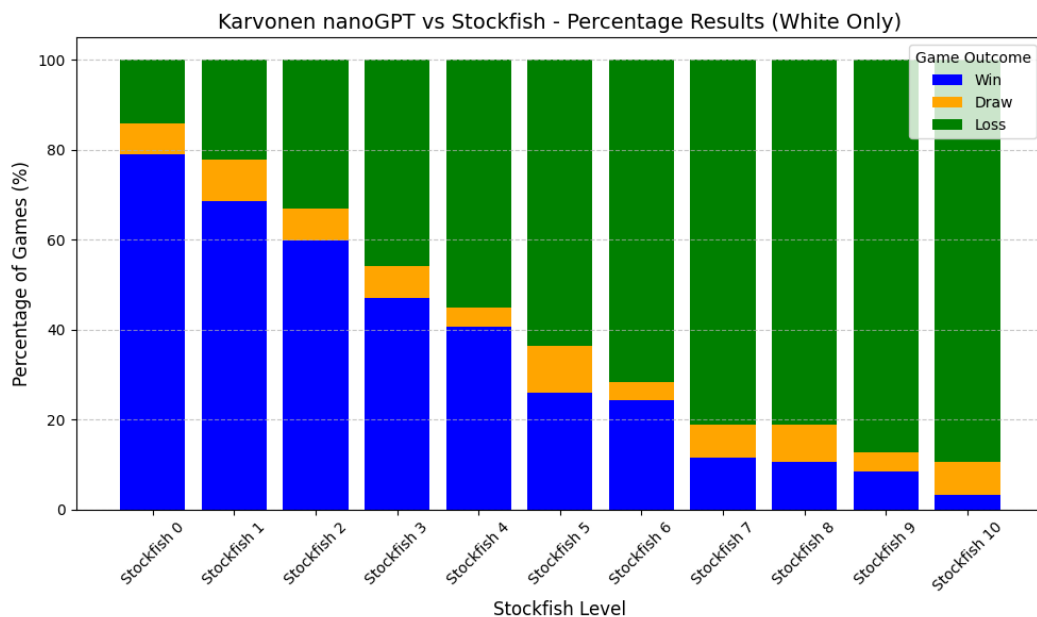
18

Figure 12: Karvonen wins out of 1000 against Stockfish 0–10 (White games only).

Noever et al.'s GPT-2 model failed to complete any games against Stockfish 0 (Elo 1300), losing every game due to repeated illegal moves. We ascribe this to short training (33,000 steps for 774M parameters), lack of tokenizer modifications, and instability in its published training loss curve.

Karvonen's nanoGPT showed strong performance as White but lost all games as Black. This asymmetry stems from dataset bias, since training games involved a strong engine always playing as White. Its estimated Elo is 2000 (White-only) or 1301 (overall), consistent with Karvonen's reported results.

Ruoss et al.'s model achieved an Elo rating of 2682, marginally higher than our Llama. It substantially outperformed against Stockfish 5–7, but converged with our model's performance at Stockfish 9–10. Their published result was 2895, but we only trained for 200K steps vs. their 1M, explaining the gap.

I.5   Interpretation

While these Elo ratings provide a relative benchmark against calibrated Stockfish levels, they are not directly comparable to official FIDE Elo ratings for human players. Nonetheless, they place our model's performance in a range typically associated with grandmaster-level play.

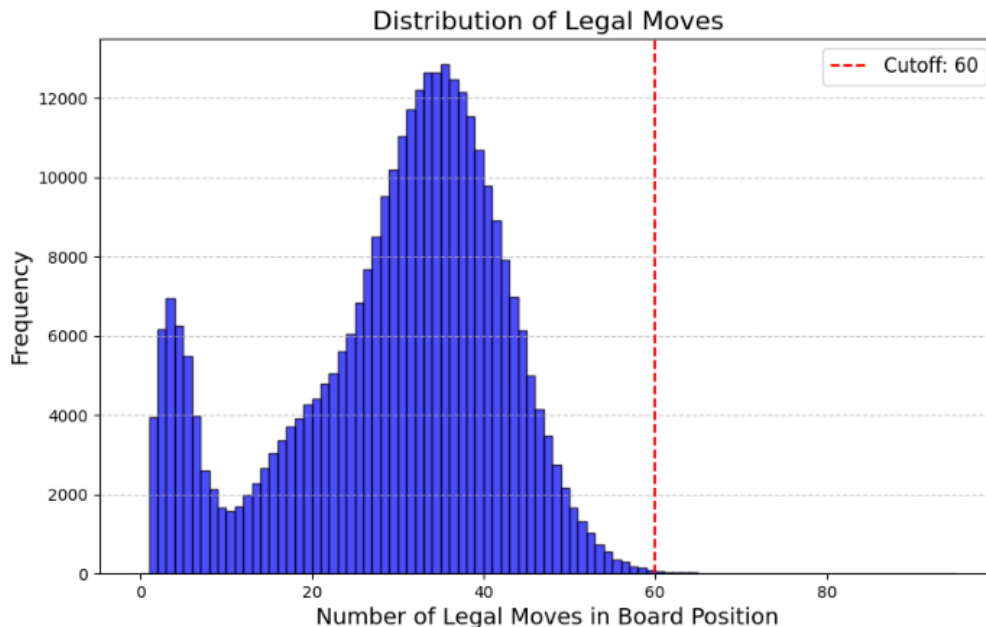# J Distributional analysis of number of legal moves in board position

Figure 13: Distribution of number of legal moves in a sample of our dataset. The secondary peak along the left tail of the distribution is due to positions in which the king is in check and other pieces may not move.

K Zero Shot Prompting Ablation Study

K.1 Prompting Ablations

Prompting strategy significantly affected zero-shot performance. Prompts framing the model as a "chess grandmaster" and constraining outputs to the provided legal moves yielded the best results. Full ablation results, prompt templates, and comparative figures are included in Appendix K.

Top 6 Performing Prompts (ordered according to name in charts above, not according to performance):

```
Prompt 1 = "<|begin_of_text|> You are a chess grandmaster.  This is the board
position in FEN notation:  {FEN tokens}.  The legal moves are:  {List of
Legal Moves in SAN}.  Which of these is the best move?  Best move:  {Best
Move in SAN} <|end_of_text|> <|pad|> <|pad|> <|pad|> <|pad|> ..."

Prompt 2 = "<|begin_of_text|> [White 'Magnus Carlsen'] [Black 'Stockfish']
Board position:  {FEN tokens}, Legal Moves:  {List of Legal Moves in SAN},
Best Move:  {Best Move} <|end_of_text|> <|pad|> <|pad|> <|pad|> <|pad|>
..."

Prompt 3 = "<|begin_of_text|> You are a chess grandmaster.  This is the
board in fen (Forsyth-Edwards notation).  It is your move:  {FEN tokens}.
Please select the best move from this list:  {List of Legal Moves in SAN}..
Please ONLY PLAY MOVES LISTED HERE. ANY move not in here is illegal.  Best
move:  {Best Move in SAN} <|end_of_text|> <|pad|> <|pad|> <|pad|> <|pad|>
..."

Prompt 4 = "<|begin_of_text|> You are analyzing a competitive chess game.  The
current board position is represented in FEN notation:  {FEN tokens}.  The
legal moves available are:  {List of Legal Moves in SAN}..  Based on the
position, decide which move is the best.  Best move:  {Best Move in SAN}
<|end_of_text|> <|pad|> <|pad|> <|pad|> <|pad|> ..."
```
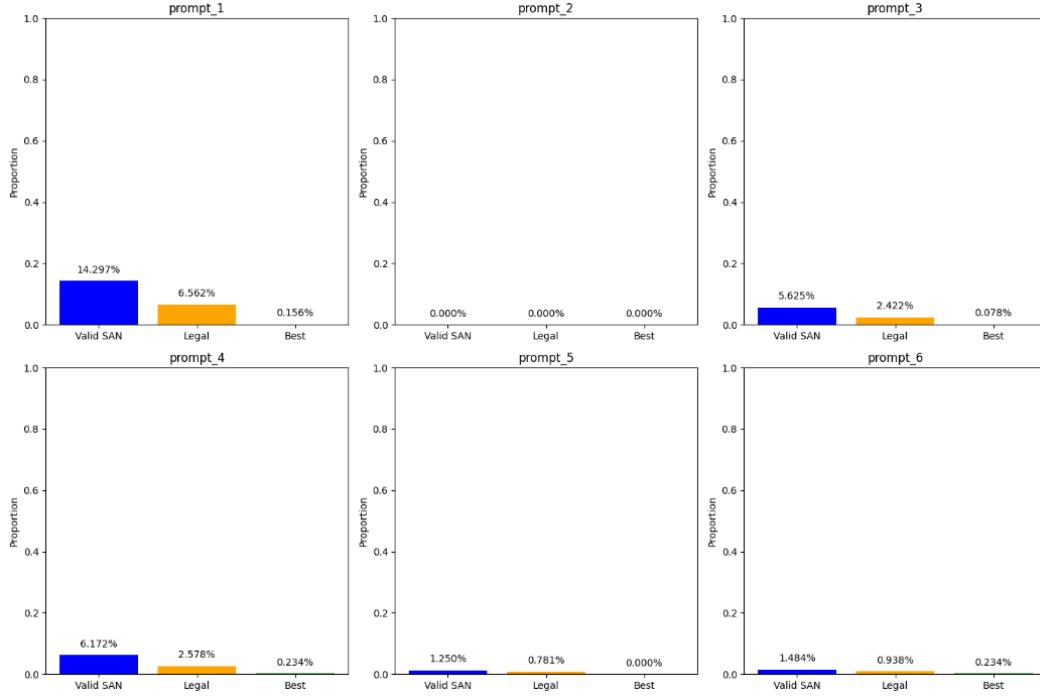
Figure 14: Baseline performance for the six best prompts (Pythia). Valid SAN denotes exemplars where Pythia output a move that could have been legal on any board state; Legal denotes exemplars where the move was legal in the current board state; Best denotes exemplars where the move matched the engine's best move.
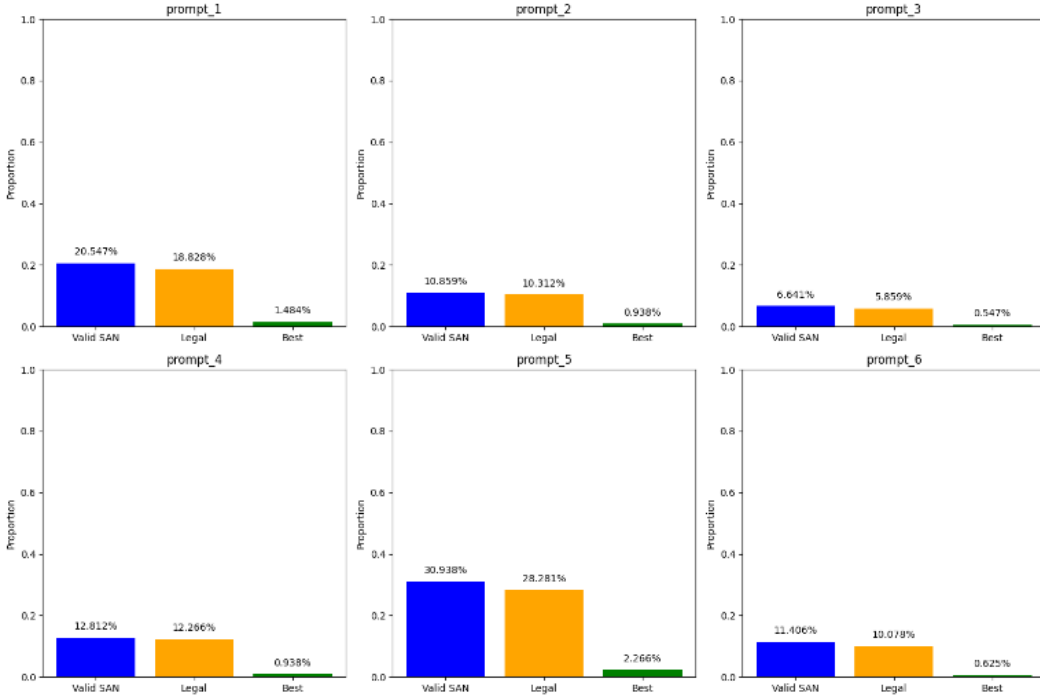

Figure 15: Baseline performance for the six best prompts (Llama).

```
Prompt 5 = "<|begin_of_text|> [FEN '{FEN tokens}'] Legal Moves: {List of
Legal Moves in SAN}. Based on the current board, determine the best move
from the provided options. Best Move: {Best Move in SAN} <|end_of_text|>
<|pad|> <|pad|> <|pad|> <|pad|> ..."
```

NeurIPS Paper Checklist

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 state two central claims: (i) that causal masking applied to spatial data is a viable and in some cases preferable training procedure for unimodal LLMs, and (ii) that this is empirically supported in the domain of chess. Both claims are matched by experiments in the paper and scoped appropriately, with limitations noted.

2. **Limitations**

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

   Answer: [Yes]

   Justification: The Limitations section covers Stockfish-calibrated Elo, threefold-repetition blindness for FEN, single-run-per-condition, and scope confined to chess. We additionally note time/resource constraints as the reason we did not conduct multi-seed variance studies.

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: No new formal theorems/proofs; we provide formalizations and intuitive arguments only.

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: The paper discloses dataset composition (FEN vs PGN), tokenization modifications, prompting templates, model sizes, and training configurations. Appendices include hyperparameter tables, compute details, and training loss curves. This information is sufficient for reproduction, and we can release code for replication upon acceptance.

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: [No]

   Justification: We will release code and artifacts upon notification of acceptance, including training scripts, custom tokenizer modifications, and the evaluation harness. Datasets used (e.g., ChessBench, Stockfish) are publicly available and cited.

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: Unchanged: we report splits, optimizer and hyperparameters, and setup differences across PGN vs. FEN.

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: Each training condition was run once due to time and resource constraints; thus we do not report multi-seed variability or CIs. We note this explicitly and leave it as future work.

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: Training hardware ($2 \times$A100 80GB), step counts, and settings are in the Training Appendix.

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: The research does not involve sensitive data, human subjects, or misuse-prone models. No human subjects or sensitive data are involved; we respect dataset and software licenses.

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper discusses both positive impacts (providing methodological insight into causal masking on spatial data, establishing chess as a sandbox for interpretability research) and limitations on broader application. Potential negative impacts (misuse of strong game-playing models for disinformation or botting) are minimal and not emphasized but could be acknowledged further.

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: [NA]

    Justification: Risk is minimal; strong chess engines (e.g., Stockfish) already exist and are widely available. No special safeguards beyond license compliance and usage notes are necessary.

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: All existing datasets and tools are cited (ChessBench, Stockfish, LLaMA, Pythia). Each has open licensing (CC or GPL for Stockfish). Licenses are respected.

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 introduce new assets (training scripts, custom tokenizer configs, evaluation harness). These will be released upon acceptance with documentation and a model card.

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: No human subjects or crowdsourcing.

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: Not applicable.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [Yes]

    Justification: LLMs (LLaMA 3.1, Pythia) are central to the methodology. Additionally, we used an LLM as a writing critic, to resolve LaTeX formatting issues, and to help troubleshoot coding issues.