

Iterative Inference in a Chess-Playing Neural Network

Elias Sandmann*
Fraunhofer HHI

Sebastian Lapuschkin
Fraunhofer HHI

Wojciech Samek
Fraunhofer HHI

Abstract

Do neural networks build their representations through smooth, gradual refinement, or via more complex computational processes? We investigate this by extending the logit lens to analyze the policy network of Leela Chess Zero, a superhuman chess engine. We find strong monotonic trends in playing strength and puzzle-solving ability across layers, yet policy distributions frequently follow non-smooth trajectories. Evidence for this includes correct puzzle solutions that are discovered early but subsequently discarded, move rankings that remain poorly correlated with final outputs, and high policy divergence until late in the network. These findings contrast with the smooth distributional convergence typically observed in language models.

1 Introduction

How do neural networks progressively build understanding as information flows through their layers? Do they incrementally refine representations by gradually increasing confidence in the correct answer, or do they fundamentally recompute preferences at each layer? Theoretical work has formalized the view that residual networks perform iterative inference, effectively implementing gradient descent in activation space—empirically resulting in higher accuracy across layers when intermediate representations are decoded through the model’s classifier (Jastrzębski et al., 2018). The logit lens (nostalgebraist, 2020) applies this approach to transformer-based language models, projecting hidden states through the unembedding matrix. This reveals similar dynamics with each layer achieving systematically lower perplexity, typically accompanied by smooth prediction trajectories showing progressively decreasing divergence from final outputs (Belrose et al., 2023; Lad et al., 2025). However, it remains unclear how iterative inference manifests empirically in other domains or for other losses and whether smooth distributional convergence is a universal property or specific to language modeling.

We examine this question by extending the logit lens to analyze the policy network of Leela Chess Zero (Leela Chess Zero team), an open-source AlphaZero (Silver et al., 2018) reimplementation that achieves strong play even without external search (lepned, 2024). Since the network is trained to distill MCTS search outcomes that optimize for optimal moves, we use playing strength as a proxy for loss reduction to test if the theoretical principle of iterative inference manifests as monotonic capability improvement. We find consistent strengthening across layers, yet find that the underlying policy distributions often evolve non-monotonically, revealing a more complex computational strategy than the smooth capability progression would suggest (see Figure 1 and Appendix F for additional visual examples demonstrating different convergence patterns).

Our contributions are: (1) We extend the logit lens technique to Post-LN transformer architectures, enabling interpretability analysis of previously inaccessible models. (2) We demonstrate that iterative inference in chess transformers exhibits distinct computational stages with monotonic capability improvement despite non-monotonic policy dynamics, revealing both parallel and contrasting patterns compared to language model inference. [Our code is available here.](#)

*Corresponding author: elias.sandmann@fraunhofer.hhi.de

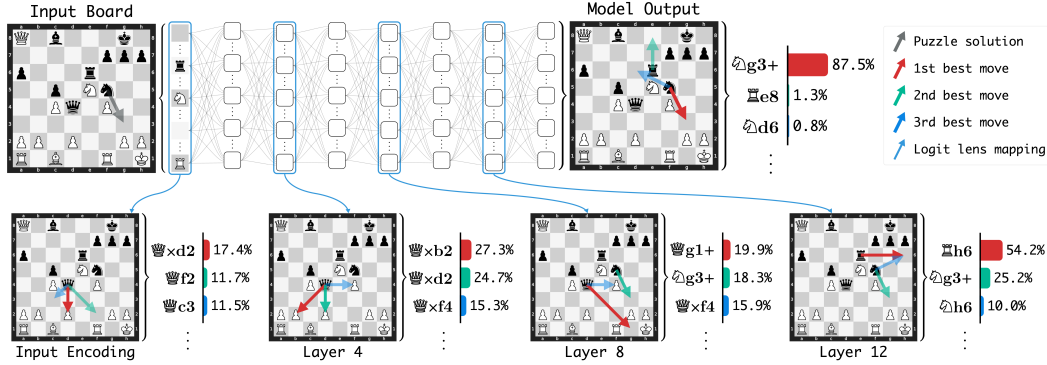


Figure 1: Our extended logit lens reveals progressive policy refinement across transformer layers in Leela Chess Zero. We map intermediate activations to policy distributions for a tactical puzzle. The model’s top-ranked move changes at each stage, with the correct solution Qg3+ only emerging as a plausible candidate in the middle layers before becoming the decisive top choice in the final output. Full probabilities are provided in Appendix E and additional examples in Appendix F.

2 Methodology

2.1 Model architecture

We analyze the T82-768x15x24h transformer model from Leela Chess Zero, the strongest neural chess engine available today (Jenner et al., 2024). This model uses a Post-LN architecture similar to the original transformer (Vaswani et al., 2017) with DeepNorm scaling (Wang et al., 2022), featuring a 15-layer transformer encoder with 768-dimensional embeddings and specialized output heads. Chess positions are encoded as 8×8 grids where each square corresponds to a token position.

Leela is trained using the AlphaZero paradigm and normally functions in tandem with MCTS as a chess engine. However, we focus solely on the policy network, which already demonstrates strong chess-playing ability without external search. Architectural details are provided in Appendix B.

2.2 Encoder-only Post-LN logit lens

The logit lens projects intermediate activations after layer ℓ through the final layer normalization and unembedding matrix to obtain layer-wise predictions. This approach works seamlessly for Pre-LN transformers, where layer normalization precedes each sublayer and leaves the residual stream free of normalization operations. Post-LN architectures create a challenge by applying normalization after residual connections instead, transforming the residual stream at each layer and creating non-linear dependencies that prevent straightforward application of the logit lens.

Functionally, the Pre-LN logit lens is equivalent to applying zero ablation to all sublayer outputs beyond a given layer ℓ (Belrose et al., 2023). We extend this principle to Post-LN models by applying the same zero ablation while preserving the subsequent layer normalization and DeepNorm scaling operations. For consistency, we also ablate layer normalization biases for layers beyond ℓ , with further justification provided in Appendix A.

Since Leela is an encoder that uses representations from all tokens simultaneously for prediction, we apply the logit lens by projecting the intermediate representations of all 64 squares through the policy head to obtain policies at each layer.

2.3 Evaluation

We follow Ruoss et al. (2024) for puzzle-solving and playing strength evaluation.

Puzzle-solving performance We evaluate tactical understanding using their chess puzzle dataset, containing 10,000 puzzles across different difficulty levels. We consider a puzzle solved if the model produces the principal variation or finds an alternative move leading to immediate checkmate. For each layer’s policy, we use argmax selection to predict moves.

Playing strength assessment We conduct a round-robin tournament between policies derived from representations at the input and all 15 transformer layers. Each pairing plays 200 games using Encyclopedia of Chess Openings (Matanović, 1978) positions with argmax move selection, totaling 27,200 games. Playing strength is computed using BayesElo with default parameters (Coulom, 2008). We include their external Leela policy network as an anchor, using their computed Elo score. Additionally, we deploy the layer-wise policies as bots on Lichess across multiple time controls, using a softmax policy with temperature 1.0 for the first five full-moves to introduce opening diversity.

Layer-wise policy metrics To analyze internal policy dynamics, we compute Jensen-Shannon divergence between layer policies and the final model output, policy entropy at each layer, the probability assigned to the final model’s top move by each layer, and Kendall’s τ ranking correlation between intermediate and final move rankings. For the ranking correlation, we also compute this metric on the GPT-2 series (Radford et al., 2019) since we have not found usage of this in the literature. Complete plots are provided in Appendix C.

3 Results

3.1 Monotonic capability progression

Puzzle solving capabilities Figure 2 shows clear improvement in puzzle-solving ability across network layers within each Elo range, and consistent decrease in performance across increasing difficulty levels for all layers. However, the gains are unevenly distributed: early and middle layers show competence on simpler puzzles but struggle significantly on harder ones, while the final layers provide the largest improvements, particularly on difficult puzzles.

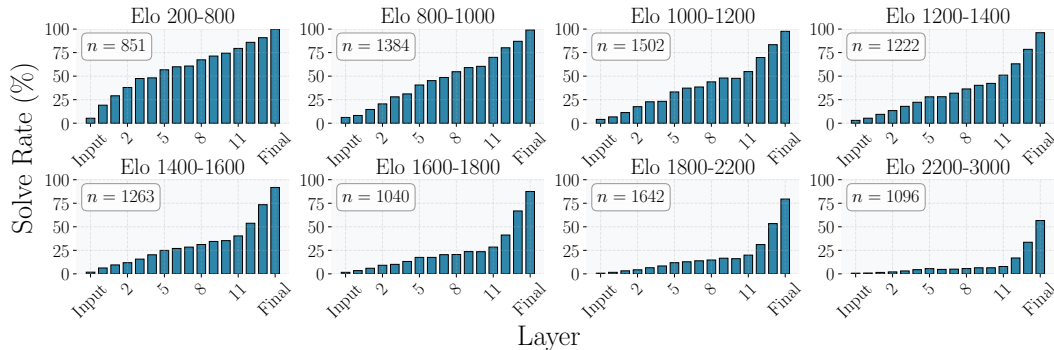


Figure 2: Puzzle-solving performance across transformer layers, stratified by Elo difficulty rating. Each subplot shows the percentage of puzzles solved by each layer for different difficulty ranges.

Tournament strength Table 1 shows overall improvement in playing strength across network depth. The internal tournament shows near-perfect monotonic Elo progression from input representation to the full model. Real-world evaluation on Lichess confirms the overall strengthening pattern, though with less perfect monotonicity.

Table 1: Playing strength across all transformer layers and evaluation methods

Evaluation	Input	L0	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	Full	Anchor
Internal Tournament	354	614	663	740	841	907	1004	1003	1016	1027	1058	1084	1121	1361	1731	2264	2292
Lichess Blitz	518	651	681	688	741	769	774	850	803	843	832	960	948	1252	1581	2274	-
Lichess Bullet	693	904	891	916	972	1009	926	984	1052	994	1061	1064	1129	1331	1659	2246	-
Lichess Rapid	558	816	709	697	915	717	939	1021	1018	1032	957	1107	1095	1290	1581	2253	-

The progression of playing strength in Table 1 is not uniform. The data suggests three distinct phases of improvement: rapid increase in the initial layers, gradual refinement through middle layers, and substantial strengthening in the final layers.

3.2 Non-monotonic policy dynamics

Puzzle solution stability While overall capability improves smoothly, individual solution finding follows a more complex process. Figure 3 tracks four metrics: current layer solutions, cumulative solutions discovered up to each layer, solutions that remain stable through all subsequent layers (converged), and newly discovered solutions (first). The gaps between these rates indicate that solutions are frequently discovered and subsequently discarded. New solution discovery is highest in initial and final layers, with lower discovery in the middle. Notably, the final cumulative solve rate exceeds the final layer’s performance, demonstrating that the complete model fails to solve some puzzles that intermediate layers handle correctly.

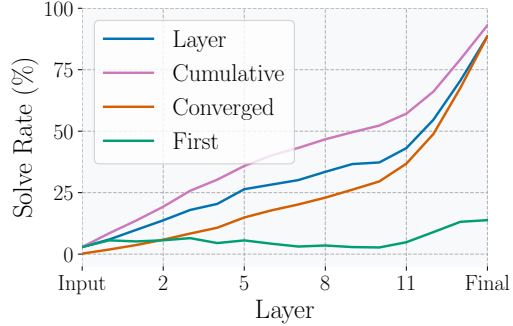


Figure 3: Layer-wise puzzle-solving performance across network depth showing raw solve rates, cumulative discoveries, converged solutions, and first discoveries.

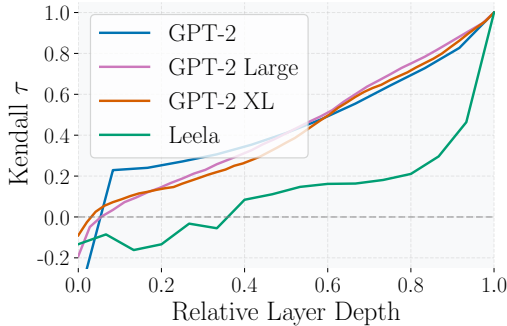


Figure 4: Median Kendall’s τ ranking correlation across network depth, comparing the policy dynamics of Leela to GPT-2 language models.

the final model’s top move also increases primarily in the final layers. Jensen-Shannon divergence shows high variability, with some positions converging early while others show high divergence until late layers. These differences may stem from distinct computational processes but could also reflect the constrained output space of chess compared to language models’ full vocabularies.

Distributional analysis Analysis of policy distributions reveals patterns distinct from the smooth convergence observed in language models (Lad et al., 2025; Belrose et al., 2023). Complete plots are provided in Appendix C. Kendall’s τ ranking correlation between intermediate and final move rankings is negative initially, remains low through the middle layers, before increasing sharply in the final layers, indicating that the model is reevaluating move preferences until very late in the network (Figure 4). This contrasts with language models, which show nearly linear improvement in ranking correlations. Policy entropy remains relatively constant across layers rather than decreasing smoothly. The probability assigned to the

4 Discussion

Our analysis provides insights into iterative inference by revealing how move preferences evolve across layers, complementing McGrath et al. (2022)’s demonstration that chess concepts are differentially represented across network depths. We observe both parallels and contrasts with language model inference. Our results show monotonic improvement in playing strength and puzzle-solving ability across layers, analogous to decreasing perplexity. However, this monotonic capability progression frequently accompanies non-monotonic policy dynamics that contrast with the smooth distributional convergence observed in language models. Like LLMs (Lad et al., 2025), chess transformers appear to exhibit distinct computational stages—middle-layer feature engineering (evidenced by plateauing strength and lower discovery rates) followed by final-layer improvements characteristic of residual sharpening (see Appendix C.1 for supporting evidence). Specifically, we observe a sharp inflection around layer 12, where all performance metrics accelerate. This aligns with Jenner et al. (2024)’s identification of the L12H12 head that relocates features from future move squares—presumably built during the preceding feature engineering phase—to immediate candidate squares, allowing the policy head to incorporate this look-ahead information into move probabilities.

Limitations A key limitation is our focus on a Post-LN architecture, making it difficult to disentangle whether our findings reflect domain-specific or architecture-specific characteristics.

References

- Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens, 2023. URL <https://arxiv.org/abs/2303.08112>.
- Rémi Coulom. Whole-history rating: A bayesian rating system for players of time-varying strength. In *Computers and Games*, 2008.
- dje dev. Discussion on layer normalization in transformer architecture. Forum post on Lc0 community chat, 2025. URL <https://lc0.org/chat>. Accessed: 2025-08-17.
- Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020. URL <https://arxiv.org/abs/2101.00027>.
- Stanisław Jastrzębski, Devansh Arpit, Nicolas Ballas, Vikas Verma, Tong Che, and Yoshua Bengio. Residual connections encourage iterative inference, 2018. URL <https://arxiv.org/abs/1710.04773>.
- Erik Jenner, Shreyas Kapur, Vasil Georgiev, Cameron Allen, Scott Emmons, and Stuart Russell. Evidence of learned look-ahead in a chess-playing neural network, 2024. URL <https://arxiv.org/abs/2406.00877>.
- Vedang Lad, Jin Hwa Lee, Wes Gurnee, and Max Tegmark. The remarkable robustness of llms: Stages of inference?, 2025. URL <https://arxiv.org/abs/2406.19384>.
- Leela Chess Zero team. Leela Chess Zero. URL <https://lczero.org/>.
- Leela Chess Zero team. A standard dataset, 2018. URL <https://lczero.org/blog/2018/09/a-standard-dataset/>. Accessed: 2025-08-20.
- lepned. How well do Lc0 networks compare to the greatest transformer network from DeepMind?, 2024. URL <https://lczero.org/blog/2024/02/how-well-do-lc0-networks-compare-to-the-greatest-transformer-network-from-deepmind/>.
- Lichess Bot Devs. Lichess bot: A bridge between lichess api and chess engines. <https://github.com/lichess-bot-devs/lichess-bot>, 2025. Accessed: 2025-08-18.
- Lichess.org. Lichess: Free online chess, 2025. URL <https://lichess.org>. Accessed: 2025-08-18.
- Aleksandar Matanović. *Encyclopaedia of Chess Openings*. Batsford Limited, 1978.
- 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), November 2022. ISSN 1091-6490. doi: 10.1073/pnas.2206625119. URL <http://dx.doi.org/10.1073/pnas.2206625119>.
- Timothee Mickus, Denis Paperno, and Mathieu Constant. How to dissect a muppet: The structure of transformer embedding spaces, 2022. URL <https://arxiv.org/abs/2206.03529>.
- Diganta Misra. Mish: A self regularized non-monotonic activation function, 2020. URL <https://arxiv.org/abs/1908.08681>.
- Daniel Monroe. Transformer progress, February 2024. URL <https://lczero.org/blog/2024/02/transformer-progress/>. Accessed: 2025-08-17.
- nostalgebraist. Interpreting gpt: the logit lens, 2020. URL <https://www.lesswrong.com/posts/AcKRB8wDpdAN6v6ru/interpreting-gpt-the-logit-lens>.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. <https://openai.com/research/language-unsupervised>, 2019. OpenAI Blog.

- Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017. URL <https://arxiv.org/abs/1710.05941>.
- 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, 2024. URL <https://arxiv.org/abs/2402.04494>.
- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419): 1140–1144, 2018.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, volume 30, pages 5998–6008, 2017.
- Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers, 2022. URL <https://arxiv.org/abs/2203.00555>.

A Post-LN Logit Lens Extension

The logit lens technique projects intermediate layer representations through the model’s output head to examine what the network would predict at different depths. This approach relies on the assumption that representations across layers exist in a shared basis that allows meaningful projection to output space—an assumption enabled by residual connections that create additive pathways for information flow (Jastrzebski et al., 2018). In transformer architectures, architectural inductive biases further encourage information about specific tokens to remain localized to their corresponding positions throughout the network, creating a privileged basis that facilitates cross-layer interpretation (Jenner et al., 2024).

In Pre-LN transformers, this involves taking intermediate representations, applying the final layer normalization, and projecting through the unembedding matrix. This approach works because layer normalization precedes each sublayer in Pre-LN models, leaving the residual stream unchanged and maintaining representational consistency across layers.

Post-LN architectures complicate this process by applying layer normalization after each sublayer’s output is added to the residual stream. Unlike Pre-LN models where only a final normalization is needed, Post-LN models have sequential normalization operations that directly transform the residual stream at each layer. These intermediate normalizations create dependencies between layers that prevent simply taking an intermediate representation and applying only the final layer normalization and output projection, as the intermediate representation has not undergone the normalization transformations it would experience in a complete forward pass.

Our goal is to develop an extension that maps intermediate layer representations to the representational basis expected by the policy head, accounting for the normalization transformations unique to Post-LN architectures.

A.1 Pre- vs Post-LN architectures and DeepNorm

The key difference between Pre-LN and Post-LN architectures lies in when layer normalization is applied relative to the residual connections. This placement affects how representations evolve through the network and impacts the applicability of interpretability techniques.

Pre-LN In Pre-LN transformers, layer normalization is applied before each sublayer (attention and feed-forward), with the computation following the pattern:

$$\mathbf{h}'_\ell = \mathbf{h}_{\ell-1} + \text{MHA}_\ell(\text{LayerNorm}(\mathbf{h}_{\ell-1})) \quad (1)$$

$$\mathbf{h}_\ell = \mathbf{h}'_\ell + \text{FFN}_\ell(\text{LayerNorm}(\mathbf{h}'_\ell)) \quad (2)$$

where \mathbf{h}_ℓ denotes the hidden representation at layer ℓ , MHA_ℓ is the multi-head attention operation, and FFN_ℓ is the feed-forward network at layer ℓ . This design ensures that the residual stream itself is never directly transformed by normalization operations, allowing intermediate representations to be projected through the final layer normalization and output head in a straightforward manner.

Post-LN Post-LN models apply layer normalization after adding sublayer outputs to the residual stream:

$$\mathbf{h}'_\ell = \text{LayerNorm}(\mathbf{h}_{\ell-1} + \text{MHA}_\ell(\mathbf{h}_{\ell-1})) \quad (3)$$

$$\mathbf{h}_\ell = \text{LayerNorm}(\mathbf{h}'_\ell + \text{FFN}_\ell(\mathbf{h}'_\ell)) \quad (4)$$

Each layer normalization operation directly transforms the accumulated representation, creating a sequence of transformations that intermediate representations must undergo to reach the final output space.

DeepNorm Leela employs DeepNorm (Wang et al., 2022) scaling to stabilize Post-LN training, modifying the computation to include residual scaling factors:

$$\mathbf{h}'_\ell = \text{LayerNorm}(\alpha \cdot \mathbf{h}_{\ell-1} + \text{MHA}_\ell(\mathbf{h}_{\ell-1})) \quad (5)$$

$$\mathbf{h}_\ell = \text{LayerNorm}(\alpha \cdot \mathbf{h}'_\ell + \text{FFN}_\ell(\mathbf{h}'_\ell)) \quad (6)$$

where $\alpha = (2N)^{1/4} \approx 2.34$ for $N = 15$ layers. This scaling, combined with specialized weight initialization, enables stable training while preserving Post-LN’s representational advantages.

A.2 Zero ablation methodology for Post-LN architectures

The standard logit lens can be understood as performing zero ablation—setting all sublayer outputs to zero for layers beyond ℓ , then applying the final layer normalization and projection to output space. We extend this principle to Post-LN models by applying the same zero ablation of sublayer outputs while preserving the normalization operations and α that would transform these representations.

Specifically, for examining layer k , we set $\text{MHA}_\ell(\cdot) = 0$ and $\text{FFN}_\ell(\cdot) = 0$ for all $\ell > k$ during a forward pass, while preserving the α and layer normalization scaling parameters (γ). We set layer normalization biases (β) to zero for layers beyond k since we believe this maintains better consistency with the standard logit lens paradigm. The rationale for these choices will be explained in the subsection following the decomposition analysis A.4.

This approach ensures that representations from layers 1 through k undergo the same sequence of transformations they would experience in the complete network, while removing contributions from later layers. Normalization statistics (μ and σ) are recomputed during this modified forward pass to reflect the actual distribution of the truncated representations, rather than using statistics computed on the full model output. This follows the same principle as Pre-LN logit lens implementations, which directly apply the final layer normalization to the intermediate activations being analyzed.

A.3 Transformer encoder decomposition

To provide intuition for why our proposed logit lens extension makes principled choices for Post-LN architectures, we present a decomposition of the transformer encoder output and reinterpret the logit lens in terms of this decomposition. Following Mickus et al. (2022), the final representation of the encoder at layer L and token position t can be decomposed into:

$$\mathbf{h}_{L,t} = \mathbf{i}_{L,t} + \mathbf{z}_{L,t}^{\text{MHA}} + \mathbf{z}_{L,t}^{\text{FFN}} + \mathbf{b}_{L,t} - \mathbf{m}_{L,t} \quad (7)$$

$$\mathbf{i}_{L,t} = \alpha^{2L} \cdot \frac{\bigodot_{\ell=1}^L \gamma_{\ell}^{\text{MHA}} \odot \gamma_{\ell}^{\text{FFN}}}{\prod_{\ell=1}^L \sigma_{\ell,t}^{\text{MHA}} \sigma_{\ell,t}^{\text{FFN}}} \odot \mathbf{h}_{0,t} \quad (8)$$

$$\mathbf{z}_{L,t}^{\text{MHA}} = \sum_{\ell=1}^L \alpha^{2L-2\ell+1} \cdot \frac{\bigodot_{\ell'=\ell}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\prod_{\ell'=\ell}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot \tilde{\mathbf{z}}_{\ell,t}^{\text{MHA}} \quad (9)$$

$$\mathbf{z}_{L,t}^{\text{FFN}} = \sum_{\ell=1}^L \alpha^{2L-2\ell} \cdot \frac{\gamma_{\ell,t}^{\text{FFN}} \bigodot_{\ell'=\ell+1}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\sigma_{\ell,t}^{\text{FFN}} \prod_{\ell'=\ell+1}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot \tilde{\mathbf{z}}_{\ell,t}^{\text{FFN}} \quad (10)$$

$$\mathbf{b}_{L,t} = \sum_{\ell=1}^L \alpha^{2L-2\ell+1} \cdot \frac{\bigodot_{\ell'=\ell}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\prod_{\ell'=\ell}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot b_{\ell}^{\text{MHA},V} W_{\ell}^{\text{MHA},O} \quad (11)$$

$$\begin{aligned} & + \sum_{\ell=1}^L \alpha^{2L-2\ell+1} \cdot \frac{\bigodot_{\ell'=\ell}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\prod_{\ell'=\ell}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot b_{\ell}^{\text{MHA},O} \\ & + \sum_{\ell=1}^L \alpha^{2L-2\ell} \cdot \frac{\gamma_{\ell,t}^{\text{FFN}} \bigodot_{\ell'=\ell+1}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\sigma_{\ell,t}^{\text{FFN}} \prod_{\ell'=\ell+1}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot b_{\ell}^{\text{FFN},O} \\ & + \sum_{\ell=1}^L \alpha^{2L-2\ell+1} \cdot \frac{\gamma_{\ell,t}^{\text{FFN}} \bigodot_{\ell'=\ell+1}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\sigma_{\ell,t}^{\text{FFN}} \prod_{\ell'=\ell+1}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot \beta_{\ell}^{\text{MHA}} \\ & + \sum_{\ell=1}^L \alpha^{2L-2\ell} \cdot \frac{\bigodot_{\ell'=\ell+1}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\prod_{\ell'=\ell+1}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \beta_{\ell}^{\text{FFN}} \\ \mathbf{m}_{L,t} = & \sum_{\ell=1}^L \alpha^{2L-2\ell+1} \cdot \frac{\bigodot_{\ell'=\ell}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\prod_{\ell'=\ell}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot \mu_{\ell,t}^{\text{MHA}} \mathbf{1} \\ & + \sum_{\ell=1}^L \alpha^{2L-2\ell} \cdot \frac{\gamma_{\ell,t}^{\text{FFN}} \bigodot_{\ell'=\ell+1}^L \gamma_{\ell'}^{\text{MHA}} \odot \gamma_{\ell'}^{\text{FFN}}}{\sigma_{\ell,t}^{\text{FFN}} \prod_{\ell'=\ell+1}^L \sigma_{\ell',t}^{\text{MHA}} \sigma_{\ell',t}^{\text{FFN}}} \odot \mu_{\ell,t}^{\text{FFN}} \mathbf{1} \end{aligned} \quad (12)$$

where $\mathbf{h}_{0,t}$ is the initial input embedding, $\tilde{\mathbf{z}}_{\ell,t}^{\text{MHA}}$ and $\tilde{\mathbf{z}}_{\ell,t}^{\text{FFN}}$ are the raw unbiased outputs from multi-head attention and feed-forward networks at layer ℓ , and $\alpha = (2N)^{1/4}$ is the DeepNorm scaling factor. The bias terms include $b_{\ell}^{\text{MHA},V}$ (value projection bias), $W_{\ell}^{\text{MHA},O}$ (output projection matrix), $b_{\ell}^{\text{MHA},O}$ (attention output bias), and $b_{\ell}^{\text{FFN},O}$ (feed-forward output bias). Layer normalization parameters are γ_{ℓ} (learned scales), $\sigma_{\ell,t}$ (computed standard deviations), $\mu_{\ell,t}$ (computed means), and β_{ℓ} (learned biases), with superscripts indicating MHA or FFN sublayers. The notation \odot denotes element-wise multiplication.

The decomposition separates the final representation into the transformed input embedding ($\mathbf{i}_{L,t}$), accumulated sublayer contributions ($\mathbf{z}_{L,t}^{\text{MHA}}$, $\mathbf{z}_{L,t}^{\text{FFN}}$), bias terms ($\mathbf{b}_{L,t}$), and mean centering effects ($\mathbf{m}_{L,t}$). Each component is scaled by normalization parameters from subsequent layers.

A.4 Implications for the logit lens

The decomposition framework provides direct justification for our zero ablation approach. The Post-LN logit lens can be understood as truncating the summations in $\mathbf{z}_{L,t}^{\text{MHA}}$, $\mathbf{z}_{L,t}^{\text{FFN}}$, and $\mathbf{b}_{L,t}$ to include only layers $\ell \leq k$, while recomputing the normalization statistics (μ and σ) based on the modified representations. This mathematical perspective clarifies why our methodological choices are well-founded.

Preservation of scaling parameters The decomposition reveals why preserving the γ and α scaling factors is essential: all terms in the truncated representation—including the input embedding $\mathbf{i}_{L,t}$ and retained sublayer contributions—are transformed by these parameters from subsequent layers.

Removing these transformations would fundamentally alter how the truncated representation maps to the output space of the encoder, undermining the interpretability objective.

Treatment of bias terms The decomposition also illuminates our decision to ablate layer normalization biases (β). These terms appear in $\mathbf{b}_{L,t}$ alongside other bias components (attention and feed-forward biases), indicating they are structurally and functionally equivalent. For consistency with the standard logit lens principle of removing sublayer contributions beyond layer k , we ablate all of the different bias terms, including layer normalization biases. Empirically, we found that preserving versus ablating these biases produces qualitatively similar results with only nuanced differences, and we provide code to examine all configurations.

Representational alignment versus learned priors The treatment of bias terms reflects broader questions about their computational role in neural networks. Bias terms may serve as representational alignment mechanisms that bridge coordinate system differences between layers, or they may encode learned priors about the task domain. For example, the Tuned Lens approach learns affine transformations with bias terms to achieve better representational alignment between an intermediate and the final layer for Pre-LN architectures, motivated by biased outputs observed with the standard logit lens (Belrose et al., 2023). However, we hypothesize that these biased outputs could also be attributed to standard logit lens implementations preserving the final layer normalization bias.

Our approach faces similar limitations by using the policy head unchanged, which contains its own bias terms that may contribute to biased outputs if our hypothesis about layer normalization bias effects in Pre-LN models is correct. Moreover, the decomposition framework only captures directly accessible bias terms—those that can be linearly separated from the representation—while bias terms that interact with activation functions cannot be decomposed into constant additive components. Likely, bias terms serve both representational alignment and learned prior functions simultaneously and it’s not clear how to disentangle these functions empirically given current interpretability techniques.

B Complete model architecture

We analyze the T82-768x15x24h transformer model from Jenner et al. (2024), using their inference framework for our experiments. This model has 15 transformer layers, 768-dimensional representations, 24 attention heads, and approximately 109M parameters. We use the original model that incorporates historical board information rather than their fine-tuned version to avoid potential artifacts from the fine-tuning process that could affect our interpretability analysis.

This Leela Chess Zero model employs a transformer-based architecture that processes chess positions through three main stages: input encoding transforms board states into token representations (with each of the 64 squares treated as a discrete token), a 15-layer transformer encoder with specialized attention mechanisms processes these representations, and task-specific output heads generate move probabilities, position evaluations, and game length predictions. The architecture employs various activation functions: Mish (Misra, 2020) for input processing and output heads, squared ReLU for feed-forward networks, and Swish (Ramachandran et al., 2017) within the Smolgen attention enhancement.

B.1 Input encoding

Leela’s input encoding transforms chess positions through binary feature planes, chess-specific positional encodings, and learned projections into the model’s embedding space.

Board representation The board state is encoded using 112 binary feature planes of size 8×8 , with the first 12 planes representing current piece positions for each piece type and color. Historical context is incorporated through 8 previous board positions (96 planes), plus auxiliary planes encoding castling rights (4 planes), side to move (1 plane), fifty-move clock (1 plane), and two constant planes (0s and 1s) that are architectural remnants from CNNs without functional meaning.

Positional encoding Leela employs chess-specific positional encodings that capture movement relationships between squares in a 64×64 matrix. Each square receives a 64-dimensional vector

where position (i, j) is set to 1 if any piece could legally move from square i to square j in one move regardless of current board state, 0 otherwise, and -1 for diagonal entries ($i = j$) to distinguish self-reference.

Input preparation The 112 feature planes are reshaped from $112 \times 8 \times 8$ to 64×112 , where each of the 64 entries corresponds to a board square with the 112-dimensional feature vector for that square concatenated with its corresponding 64-dimensional positional encoding vector, forming a 64×176 tensor. This combined representation undergoes a linear transformation with Mish activation, followed by elementwise scaling and shifting operations, producing the final 64×768 input for the transformer layers. This enriched preprocessing was motivated by observations that early attention layers contributed minimally to performance (Monroe, 2024). While no information mixing between tokens occurs here, this involves substantially more processing than typical language models (which simply use embedding matrices plus positional encodings), potentially explaining why our logit lens mappings yield meaningful results even before the first transformer layer.

B.2 Transformer encoder

The transformer encoder consists of 15 identical layers with a model dimensionality of 768, processing 64 tokens (one per board square) through multi-head attention with Smolgen enhancement, feed-forward networks, and Post-LN normalization with DeepNorm scaling. Unlike autoregressive models, all tokens can attend to each other bidirectionally.

Attention with Smolgen Each layer uses 24 attention heads with 32-dimensional head size, applying scaled dot-product attention to the 64 board square tokens. The Smolgen module enhances standard self-attention by enabling attention scores to depend not only on individual square contents but also on the global board position. It compresses all square representations into a global vector, processes this through MLPs, then generates supplementary attention logits of shape $24 \times 64 \times 64$ that are added to the standard query-key dot products before softmax normalization.

Feed-Forward Each FFN layer expands from 768 to 1024 dimensions with squared ReLU activation. Unlike other domains that benefit from $4\times$ expansion ratios, chess models show little improvement from larger feed-forward networks (Monroe, 2024).

Post-LN with DeepNorm The model employs a Post-LN architecture, the original design used in early transformers (Vaswani et al., 2017), applying layer normalization after each sublayer (attention and feed-forward) within the residual stream rather than before sublayers as in modern Pre-LN variants, following empirical comparisons that demonstrated superior performance (djee dev, 2025). To mitigate vanishing gradients, the model uses DeepNorm scaling (Wang et al., 2022), which applies a constant upscaling factor to residual connections and uses specialized weight initialization. See Appendix A.1 for complete equations.

B.3 Output heads

The transformer encoder’s final 64×768 representations are processed by three specialized heads: the policy head generates move probability distributions, the value head predicts win/draw/loss outcomes, and the moves-left head estimates remaining game length.

Policy head The policy head first processes each square’s 768-dimensional representation through a shared MLP with Mish activation. These processed representations are then transformed via two separate linear projections: one creating “source” representations for squares where moves originate, and another creating “target” representations for destination squares. The source and target representations are matrix-multiplied along the 768-dimensional axis, producing a 64×64 matrix where each entry contains a scalar logit representing the likelihood of a move from the corresponding source square to target square. Promotion moves require additional processing through two specialized branches that handle move selection and promotion-type preferences separately. The final output combines standard move logits with promotion logits, which are then filtered to extract only legal moves for the current position before applying softmax normalization.

Value head The value head processes each square’s representation through an MLP with Mish activation, reducing them to 32 dimensions per square. These 64 representations are flattened into a single 2048-dimensional vector to enable global position assessment. A second MLP layer compresses this to 128 dimensions, followed by a linear projection to 3 logits, which are then converted to win, draw, and loss outcome probabilities via softmax normalization. Unlike AlphaZero’s single scalar output, this three-way classification provides more nuanced position evaluation.

Moves-left head The moves-left head predicts the number of moves remaining until game termination. Each square’s representation is processed through an MLP, then flattened into a global vector for position-level assessment. Two additional MLP layers progressively reduce the dimensionality to produce a final scalar output estimating remaining game length. The output layer uses Mish activation because ReLU would provide zero gradients when pre-activation values are negative during training, even though the target output is always positive.

C Detailed policy dynamics analysis

To provide a deeper, quantitative view of the model’s inference process, we computed several policy metrics across the network’s depth. The following analyses were conducted on a sample of 1000 positions from the CCRL dataset (Leela Chess Zero team, 2018). For the language model comparisons, we sampled prompts from the Pile dataset (Gao et al., 2020).

C.1 Policy Metrics

Figure 5 presents four metrics that characterize the policy’s evolution. We plot the median, the 25th-75th percentile range, and the 5th-95th percentile range.

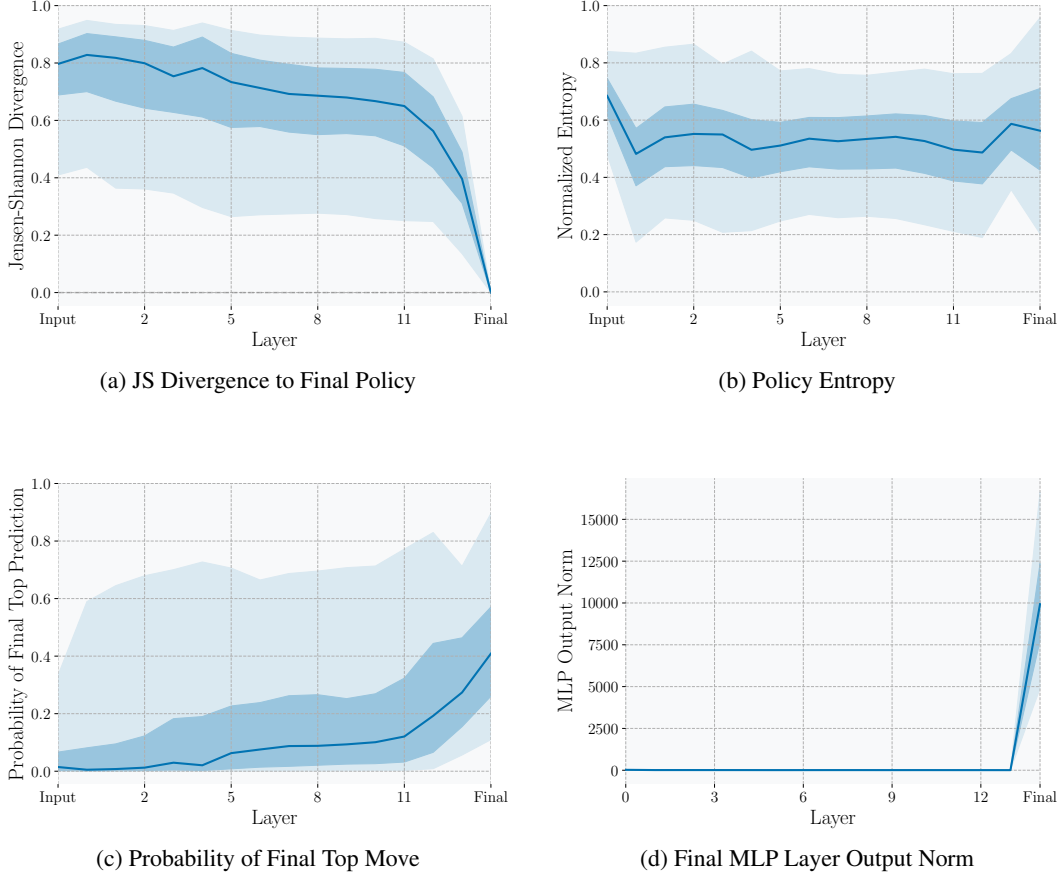


Figure 5: Metrics of policy dynamics for Leela Chess Zero. The central line shows the median over 1000 positions. The inner (darker) and outer (lighter) shaded regions represent the 25th-75th and 5th-95th percentile ranges, respectively.

Jensen-Shannon divergence The Jensen-Shannon Divergence shows a median that remains high until the final layers, with the 5th percentile being much lower. This shows that a small subset of positions converge early to the final policy, while the vast majority have different move preferences than the final output throughout most layers.

Entropy The policy entropy shows a relatively constant median uncertainty, meaning all layers are more or less equally certain about what moves they prefer (they just prefer different moves). The 95th and 5th percentiles are very high and very low respectively, indicating that for any layer there are positions where the layer is either very certain or very uncertain about what moves are good.

Probability of final top move The median probability assigned to the final top move is near zero for the first layers, then grows linearly with a steeper increase in the final layers. However, the outliers

are very high (between 0.6 and 0.8), indicating that all layers correctly predict the top move for some positions with high probability, either having already converged or representing cases where solutions are later forgotten.

MLP output norms Following [Lad et al. \(2025\)](#), we also examine the L2 norm of the final MLP layer’s output to probe for their proposed residual sharpening mechanism. We observe a similar but more extreme pattern, with sharply increasing MLP norms in the last layer, consistent with their findings in language models.

C.2 Ranking Correlation (Kendall’s τ) Analysis

Chess models To analyze the stability of move preferences, we compute the Kendall’s τ rank correlation between intermediate and final policies (Figure 6). We calculate this metric in two ways: first using all legal moves, and second, to mitigate noise from low probability moves that are never seriously considered, we only calculate over moves that appear in the top 5 at any layer. For both approaches, the correlation is negative in early layers and remains low until around the 12th layer, where it increases sharply. The 5th and 95th percentiles in the top-5 moves analysis show that there are positions where intermediate layer move rankings are either very similar or very dissimilar to the final ranking until late in the network.

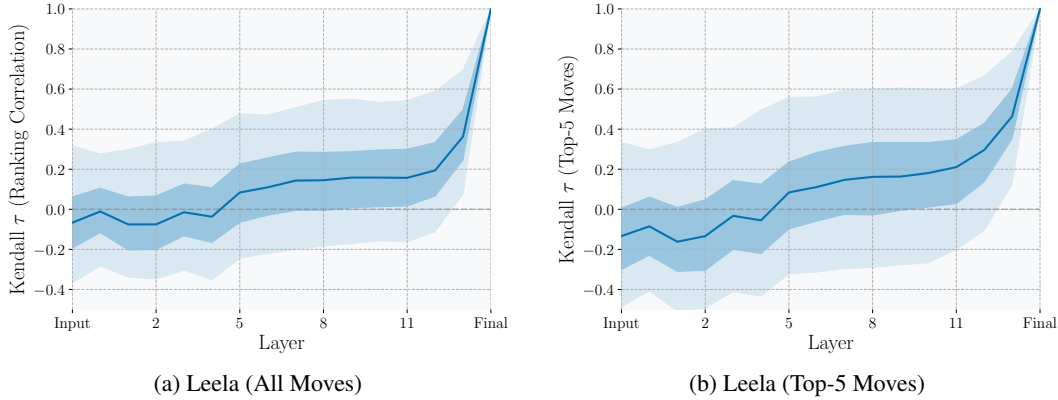


Figure 6: Kendall’s τ move ranking correlation for Leela Chess Zero. The shaded regions represent the 5th-95th and 25th-75th percentile ranges.

Language models Since we have not found analysis with this metric for language models, we conducted our own experiments for comparison using the GPT-2 series (small, large, XL) ([Radford et al., 2019](#)). To mitigate noise from low probability tokens, we only used tokens that appear in the top 100 predictions at any layer (gathering all top 100 tokens across all layers). The plots show a nearly linear increase in correlation with a notable jump from the input embedding to the 0th layer, which is expected since the input embedding represents only the pure embedding of the previous token.

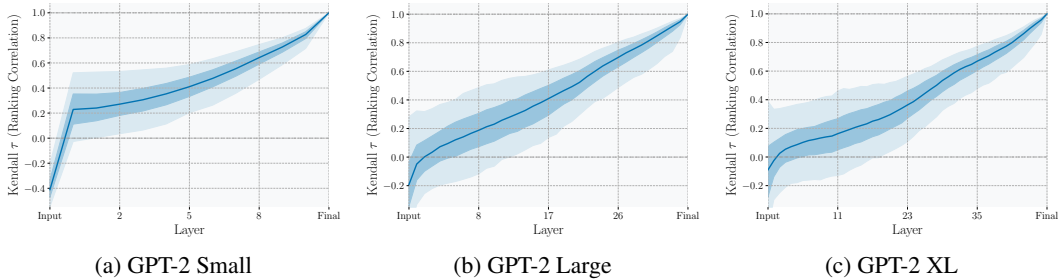


Figure 7: Kendall’s τ token ranking correlation for GPT-2 models. The shaded regions represent the 5th-95th and 25th-75th percentile ranges.

This suggests a difference in computational strategies between chess and language models, with chess models extensively reordering move preferences throughout the network while language models show more consistent ranking evolution.

D Complete tables for the tournament and Lichess play

D.1 Internal Tournament

The complete tournament results processed with BayesElo (Coulom, 2008) using the default confidence parameter of 0.5 are presented in Table 2. The computed Elo ratings demonstrate a clear progression of playing strength across layers. The table shows each model’s Elo rating with asymmetric confidence intervals (+ and - columns), the percentage of points scored (wins + 0.5×draws), the average Elo of opponents faced, and the draw rate. All models played exactly 3200 games in the round-robin tournament format.

Table 2: Playing strength evaluation through internal tournament results

Rank	Model	Elo	+	-	Games	Score (%)	Avg. Oppo.	Draws (%)
1	External Lc0 Model	2292	35	34	3200	97	1049	3
2	Full Model	2264	34	34	3200	96	1051	3
3	Logit Lens Layer 13	1731	33	32	3200	86	1084	2
4	Logit Lens Layer 12	1361	18	18	3200	74	1108	5
5	Logit Lens Layer 11	1121	14	14	3200	58	1123	9
6	Logit Lens Layer 10	1084	13	13	3200	55	1125	13
7	Logit Lens Layer 9	1058	13	13	3200	53	1126	14
8	Logit Lens Layer 8	1027	13	13	3200	50	1128	14
9	Logit Lens Layer 7	1016	13	13	3200	49	1129	15
10	Logit Lens Layer 5	1004	12	12	3200	49	1130	21
11	Logit Lens Layer 6	1003	13	13	3200	48	1130	14
12	Logit Lens Layer 4	907	12	12	3200	39	1136	19
13	Logit Lens Layer 3	841	13	13	3200	33	1140	18
14	Logit Lens Layer 2	740	13	14	3200	24	1146	17
15	Logit Lens Layer 1	663	14	15	3200	18	1151	16
16	Logit Lens Layer 0	614	16	16	3200	15	1154	9
17	Logit Lens Input	354	23	25	3200	4	1170	4

D.2 Lichess bot performance

To validate our internal tournament findings in a real-world setting, we deployed layer-wise policies as bots on Lichess ([Lichess.org](https://lichess.org), 2025) using the bot API framework (Lichess Bot Devs, 2025). Due to rate-limiting constraints, we distributed bots across multiple cloud instances and used policy sampling for the first five moves to introduce opening diversity. The bots participated in Bullet (1+0, 2+1), Blitz (3+0, 3+2, 5+0, 5+3), and Rapid (10+0, 10+5, 15+10) time controls until ratings stabilized. Several experimental constraints shaped the evaluation: (1) bots played exclusively against other bots per Lichess policy, (2) the limited pool of weak bots meant repeated matchups for early layers, and (3) deterministic play after the opening often led to repetitive game. Despite these limitations, the results largely corroborate our internal tournament findings, showing progressive improvement across layers with the most significant transitions occurring at similar points in the network depth. Table 3 presents the playing strength and performance statistics for our layer-wise Lichess bots, with bot names linked to their respective Lichess profiles.

Table 3: Performance of layer-wise policies on Lichess across different time controls

Bot	Rating			Total Games	Performance		
	Bullet	Blitz	Rapid		W	D	L
LLLBot-In	693 \pm 54	518 \pm 45	558 \pm 45	437	2	150	285
LLLBot-0	904 \pm 54	651 \pm 45	816 \pm 50	419	2	85	332
LLLBot-1	891 \pm 54	681 \pm 45	709 \pm 46	431	5	92	334
LLLBot-2	916 \pm 50	688 \pm 45	697 \pm 45	453	13	134	306
LLLBot-3	972 \pm 55	741 \pm 45	915 \pm 50	463	9	80	374
LLLBot-4	1009 \pm 55	769 \pm 45	717 \pm 45	461	12	136	313
LLLBot-5	926 \pm 56	774 \pm 45	939 \pm 51	468	23	66	379
LLLBot-6	984 \pm 64	850 \pm 45	1021 \pm 46	446	22	62	362
LLLBot-7	1052 \pm 55	803 \pm 45	1018 \pm 49	452	23	50	379
LLLBot-8	994 \pm 63	843 \pm 45	1032 \pm 47	450	23	58	369
LLLBot-9	1061 \pm 58	832 \pm 45	957 \pm 54	445	32	47	366
LLLBot-10	1064 \pm 59	960 \pm 45	1107 \pm 45	494	39	73	382
LLLBot-11	1129 \pm 54	948 \pm 45	1095 \pm 45	436	34	46	356
LLLBot-12	1331 \pm 49	1252 \pm 45	1290 \pm 45	415	74	63	278
LLLBot-13	1659 \pm 48	1581 \pm 45	1581 \pm 45	368	124	39	205
LLLBot-Full	2246 \pm 52	2274 \pm 45	2253 \pm 52	316	197	32	87

E Complete probabilities for the example puzzle from Figure 1

The puzzle (Figure 8) features a knight sacrifice leading to mate (PV: 1... ♘g3+ 2. h×g3 ♖h6#). The input layer exhibits piece-specific bias toward queen moves, with multiple queen captures dominating early probabilities. The winning move ♘g3+ first becomes the top choice at layer 5 (21.09%) but is then overtaken by ♖g1+, which dominates layers 6-10. This temporary preference for ♖g1+ likely reflects a learned heuristic of the model that queen checks are tactically important, even though this particular check ultimately loses the queen without compensation. After layer 10, ♘g3+ regains its position as the preferred move, with the exception of layer 12 where a rook lift ♖h6 (54.15%) briefly becomes the top candidate, potentially pinning the h-pawn and threatening the king. The winning move's probability follows a non-monotonic trajectory, fluctuating throughout the layers before ultimately surging to 47.38% at layer 13 and 87.55% in the final output. Despite these fluctuations, this example demonstrates a relatively smooth progression compared to other cases in Appendix F, where the correct move is either identified immediately after early layers or remains unconsidered until the very late layers. Full probabilities are in Table 4.

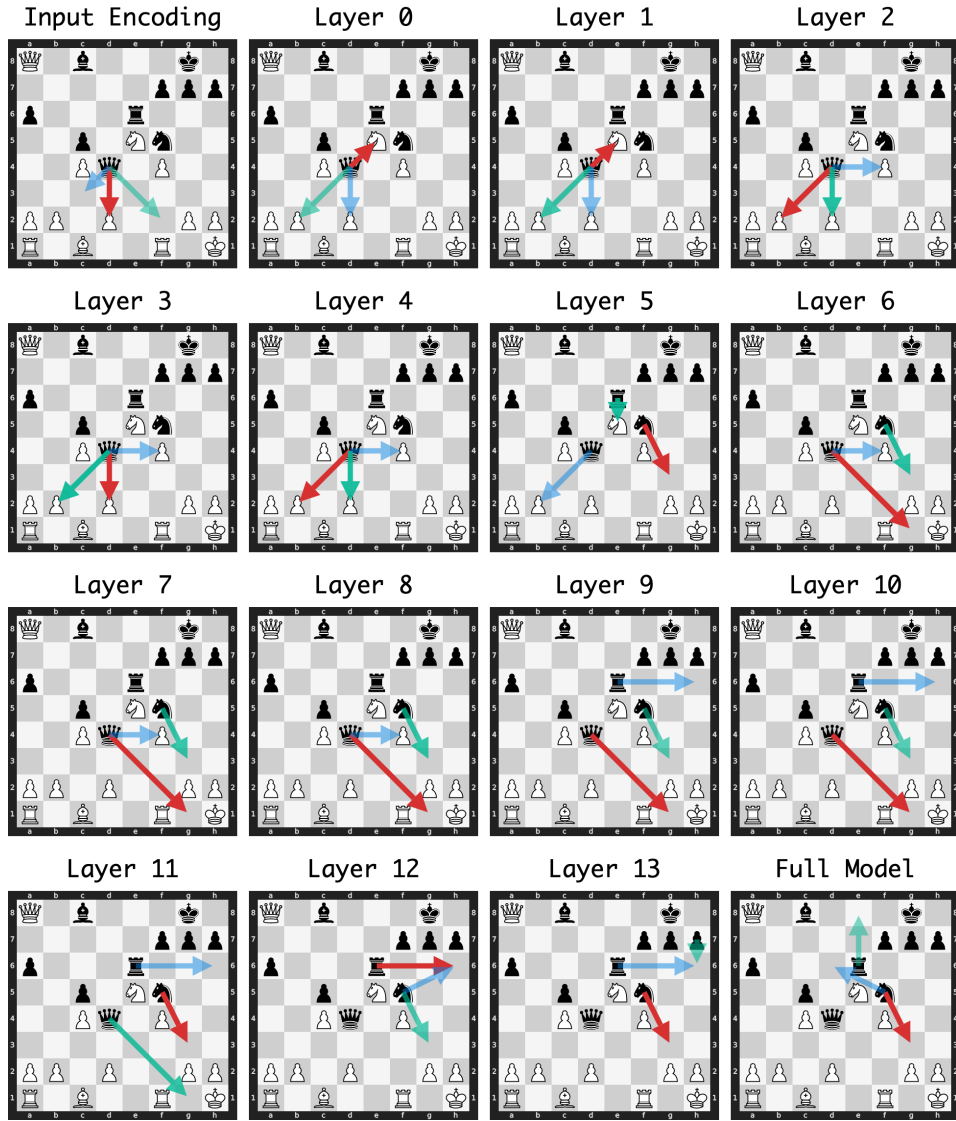


Figure 8: Layer-wise policy evolution for the puzzle in Figure 1.

Table 4: Layer-wise policy probability evolution (Part 1: Input to Layer 6)







































Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
 g3+	3.70%	0.04%	0.20%	2.30%	8.94%	0.62%	21.09%	24.77%
 h6	0.01%	0.01%	0.03%	0.10%	0.11%	0.04%	0.03%	3.50%
 g1+	6.51%	1.65%	5.34%	8.90%	8.76%	1.06%	2.94%	29.18%
 xb2	5.96%	17.19%	20.98%	24.84%	22.25%	27.29%	13.75%	3.22%
 xe5	1.26%	27.24%	21.76%	3.97%	2.18%	2.71%	2.28%	0.85%
 xd2	17.39%	15.37%	20.20%	22.17%	24.44%	24.74%	12.49%	2.41%
 xe5	0.19%	13.28%	13.17%	8.52%	5.92%	11.72%	19.41%	8.12%
 h6	0.01%	1.16%	0.01%	0.01%	0.00%	0.01%	0.00%	0.00%
 xf4	4.20%	7.53%	7.52%	15.33%	12.42%	15.28%	13.52%	16.19%
 xc4	4.38%	10.67%	7.54%	11.15%	12.02%	12.68%	13.12%	8.54%
 f2	11.68%	0.11%	0.69%	0.08%	0.16%	0.06%	0.02%	0.09%
 c3	11.47%	0.03%	0.09%	0.49%	0.28%	0.28%	0.23%	0.11%
 h6	0.09%	0.05%	0.01%	0.03%	0.00%	0.02%	0.03%	1.90%
 d3	9.82%	0.02%	0.23%	0.05%	0.29%	0.04%	0.05%	0.08%
 e3	9.55%	0.06%	0.07%	0.17%	0.06%	0.07%	0.06%	0.12%
 e4	5.76%	0.13%	0.28%	0.03%	0.07%	0.03%	0.04%	0.05%
 e3	4.74%	0.05%	0.10%	0.29%	0.10%	0.11%	0.04%	0.02%
 h5	0.02%	2.46%	0.11%	0.03%	0.02%	0.01%	0.01%	0.01%
 d6	0.07%	0.01%	0.02%	0.05%	0.16%	0.03%	0.04%	0.05%
 e8	0.01%	0.80%	0.36%	0.08%	0.03%	0.01%	0.01%	0.02%
 h8	0.39%	0.03%	0.02%	0.11%	0.56%	1.32%	0.16%	0.14%
 f6	0.13%	0.02%	0.00%	0.00%	0.04%	0.01%	0.02%	0.05%
 c6	0.20%	0.84%	0.47%	0.41%	0.32%	0.66%	0.21%	0.16%
 d8	0.05%	0.15%	0.04%	0.21%	0.06%	0.05%	0.02%	0.02%
 g6	0.10%	0.05%	0.01%	0.04%	0.03%	0.02%	0.02%	0.01%
 e7	0.25%	0.00%	0.01%	0.03%	0.02%	0.03%	0.02%	0.02%
 g5	0.78%	0.18%	0.19%	0.04%	0.02%	0.02%	0.01%	0.01%
 a5	0.01%	0.75%	0.15%	0.18%	0.05%	0.04%	0.03%	0.01%
 d6	0.05%	0.01%	0.02%	0.03%	0.02%	0.04%	0.02%	0.02%
 f8	0.07%	0.03%	0.03%	0.03%	0.15%	0.31%	0.06%	0.09%
 d7	0.06%	0.01%	0.01%	0.03%	0.05%	0.10%	0.02%	0.02%
 d5	0.35%	0.01%	0.02%	0.07%	0.08%	0.05%	0.11%	0.09%
 d6	0.11%	0.01%	0.03%	0.04%	0.03%	0.04%	0.01%	0.02%
 b6	0.05%	0.01%	0.01%	0.03%	0.09%	0.24%	0.04%	0.03%
 f6	0.16%	0.01%	0.17%	0.08%	0.05%	0.03%	0.01%	0.02%
 g6	0.08%	0.01%	0.02%	0.04%	0.06%	0.08%	0.01%	0.01%
 h4	0.13%	0.01%	0.01%	0.03%	0.08%	0.01%	0.04%	0.07%
 e7	0.21%	0.01%	0.07%	0.03%	0.04%	0.13%	0.02%	0.02%

Table 5: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♟g3+	22.95%	18.27%	18.48%	21.85%	33.31%	25.23%	47.38%	87.55%
♚h6	7.29%	14.22%	9.13%	10.10%	25.73%	54.15%	10.59%	0.24%
♙g1+	31.08%	19.92%	48.52%	49.22%	30.02%	3.93%	2.22%	0.28%
♙xb2	0.96%	4.92%	0.59%	0.47%	0.14%	0.06%	0.85%	0.25%
♙xe5	0.52%	0.71%	0.99%	0.30%	0.09%	0.04%	0.58%	0.37%
♙xd2	0.78%	3.62%	0.50%	0.43%	0.07%	0.04%	0.58%	0.24%
♚xe5	7.27%	5.84%	5.52%	3.57%	2.38%	1.35%	0.70%	0.37%
h6	0.05%	0.38%	0.30%	1.07%	2.08%	3.49%	17.35%	0.25%
♙xf4	13.18%	15.89%	6.21%	6.16%	0.78%	0.17%	0.48%	0.27%
♙xc4	9.54%	10.13%	4.60%	1.64%	0.36%	0.14%	0.64%	0.29%
♙f2	0.14%	0.38%	0.29%	1.37%	0.14%	0.03%	0.25%	0.27%
♙c3	0.22%	0.09%	0.05%	0.02%	0.04%	0.01%	0.46%	0.22%
♟h6	4.86%	4.45%	3.21%	2.62%	3.90%	10.01%	6.52%	0.23%
♙d3	0.05%	0.09%	0.11%	0.06%	0.11%	0.05%	0.40%	0.26%
♙e3	0.05%	0.03%	0.01%	0.01%	0.02%	0.02%	0.29%	0.25%
♙e4	0.04%	0.03%	0.02%	0.01%	0.02%	0.01%	0.24%	0.26%
♟e3	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.18%	0.21%
h5	0.09%	0.04%	0.07%	0.12%	0.10%	0.06%	0.72%	0.25%
♟d6	0.10%	0.09%	0.37%	0.17%	0.08%	0.10%	1.95%	0.82%
♚e8	0.10%	0.16%	0.20%	0.09%	0.08%	0.29%	0.94%	1.35%
♙h8	0.15%	0.15%	0.29%	0.29%	0.16%	0.28%	0.27%	0.27%
f6	0.13%	0.07%	0.06%	0.06%	0.06%	0.13%	1.02%	0.29%
♚c6	0.09%	0.05%	0.04%	0.01%	0.02%	0.02%	0.18%	0.34%
♙d8	0.03%	0.04%	0.05%	0.05%	0.04%	0.04%	0.49%	0.81%
g6	0.02%	0.02%	0.02%	0.03%	0.03%	0.05%	0.79%	0.31%
♟e7	0.01%	0.01%	0.02%	0.01%	0.01%	0.01%	0.79%	0.39%
g5	0.01%	0.01%	0.02%	0.02%	0.01%	0.01%	0.50%	0.29%
a5	0.02%	0.01%	0.01%	0.00%	0.00%	0.01%	0.14%	0.23%
♙d6	0.02%	0.02%	0.03%	0.03%	0.03%	0.03%	0.42%	0.28%
♙f8	0.06%	0.10%	0.09%	0.05%	0.02%	0.04%	0.19%	0.35%
♙d7	0.01%	0.02%	0.02%	0.02%	0.03%	0.03%	0.15%	0.35%
♙d5	0.09%	0.10%	0.08%	0.04%	0.06%	0.02%	0.34%	0.27%
♚d6	0.01%	0.02%	0.02%	0.01%	0.02%	0.02%	0.32%	0.29%
♚b6	0.02%	0.01%	0.01%	0.01%	0.01%	0.02%	0.20%	0.28%
♚f6	0.02%	0.03%	0.01%	0.02%	0.01%	0.03%	0.19%	0.28%
♚g6	0.01%	0.02%	0.01%	0.02%	0.03%	0.04%	0.17%	0.28%
♟h4	0.04%	0.05%	0.01%	0.01%	0.01%	0.03%	0.26%	0.20%
♚e7	0.01%	0.01%	0.01%	0.00%	0.01%	0.01%	0.26%	0.26%

F Puzzle Case Studies

F.1 Case Study 1: Knight Fork and Discovered Attack

This puzzle (Figure 9) features a tactical sequence beginning with a knight check that sets up a discovered attack (PV: 1. ♖d6+ e×d6 2. ♗×c6+). After the knight is captured, the bishop delivers a check, forcing the black king to move and allowing White to capture the undefended queen. The input layer exhibits piece-specific bias toward knight moves, which are abandoned after layer 0 as the model shifts focus to queen captures. The queen trade ♗×a5 dominates through most layers with the queen being protected by a knight. The winning move ♖d6+ maintains low probability until layer 12 (2.77%), then suddenly jumps to 26.23% at layer 13 before becoming the decisive top choice in the final output. This sudden increase after layer 12—the same layer Jenner et al. (2024) identified as containing a “look-ahead” head—may indicate the model requires multi-move analysis rather than simple tactical heuristics for this position. Full probabilities are in Table 6.

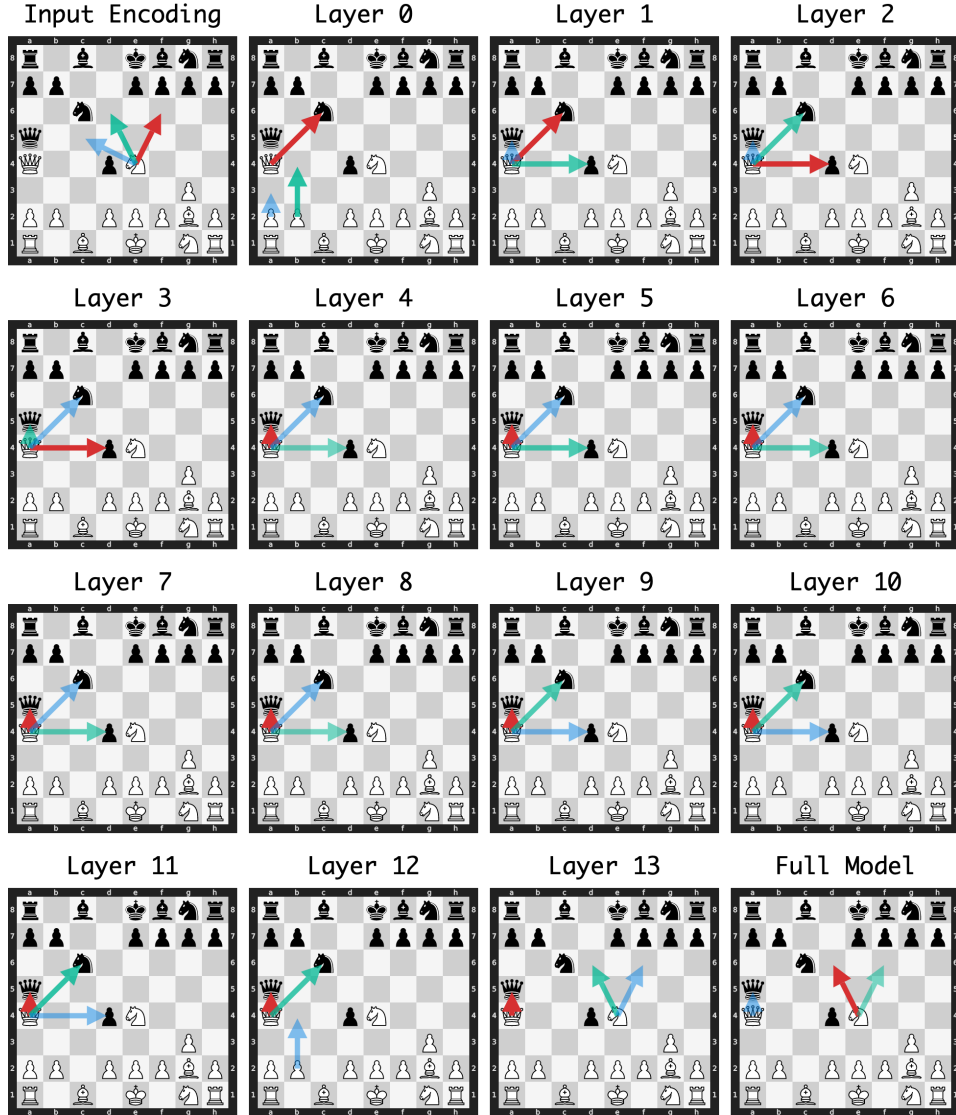


Figure 9: Layer-wise policy evolution for Puzzle 12864.

Table 6: Layer-wise policy probability evolution (Part 1: Input to Layer 6)

Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
♟d6+	21.53%	0.02%	1.32%	4.01%	0.10%	0.04%	2.37%	3.13%
♚xa5	0.04%	6.74%	7.68%	10.13%	25.17%	48.88%	44.79%	51.85%
♚xc6+	0.46%	35.02%	50.31%	24.26%	21.72%	21.79%	13.27%	10.83%
♚xd4	2.15%	11.37%	26.27%	41.95%	30.81%	23.59%	25.63%	20.43%
b4	2.28%	27.81%	0.56%	1.23%	1.40%	0.01%	0.02%	0.35%
♟f6+	23.64%	0.01%	0.15%	0.82%	0.03%	0.03%	0.01%	0.06%
♟c5	17.72%	0.02%	0.10%	0.23%	0.16%	0.06%	0.31%	0.19%
f4	17.23%	0.22%	0.03%	0.04%	0.03%	0.07%	0.04%	0.19%
a3	0.02%	11.77%	4.90%	7.07%	0.14%	0.32%	0.07%	0.15%
♟f3	0.22%	0.01%	0.05%	0.36%	5.93%	0.12%	5.70%	4.50%
g4	5.44%	0.30%	0.35%	0.88%	0.26%	0.11%	0.02%	0.01%
♚b4	0.09%	0.21%	0.61%	0.93%	4.29%	0.28%	0.11%	0.43%
♟f3	0.75%	0.01%	0.89%	0.66%	2.86%	0.92%	3.73%	2.78%
♚b3	0.09%	0.04%	0.20%	1.03%	2.62%	0.44%	1.40%	2.02%
h3	0.03%	2.71%	0.21%	0.48%	0.03%	0.08%	0.10%	0.05%
♟g5	2.46%	0.02%	0.02%	0.14%	0.06%	0.10%	0.03%	0.02%
♚a3	0.03%	0.72%	0.78%	1.16%	1.61%	0.40%	0.36%	1.05%
e3	1.36%	0.55%	0.07%	0.32%	0.17%	0.40%	0.22%	0.18%
♟f1	0.00%	0.13%	1.30%	0.55%	0.48%	0.20%	0.13%	0.06%
♟c3	1.28%	0.01%	0.93%	0.43%	0.07%	0.07%	0.03%	0.07%
♟f1	0.64%	0.52%	1.28%	0.42%	0.27%	0.11%	0.12%	0.07%
h4	0.06%	0.88%	1.27%	0.77%	0.15%	0.13%	0.09%	0.09%
b3	0.27%	0.23%	0.15%	1.25%	0.01%	0.03%	0.04%	0.02%
f3	0.80%	0.00%	0.00%	0.00%	0.01%	0.09%	0.06%	0.13%
♟h3	0.02%	0.05%	0.11%	0.13%	0.82%	0.69%	0.34%	0.20%
♚c4	0.60%	0.04%	0.02%	0.07%	0.19%	0.08%	0.17%	0.22%
♚b1	0.22%	0.08%	0.09%	0.17%	0.11%	0.34%	0.28%	0.29%
♟d1	0.20%	0.08%	0.07%	0.09%	0.18%	0.29%	0.14%	0.16%
♚d1	0.05%	0.06%	0.04%	0.08%	0.04%	0.09%	0.08%	0.05%
♚c2	0.09%	0.04%	0.04%	0.09%	0.04%	0.10%	0.11%	0.16%
♟h3	0.11%	0.27%	0.12%	0.15%	0.03%	0.07%	0.13%	0.11%
♚b5	0.11%	0.05%	0.08%	0.08%	0.23%	0.08%	0.10%	0.13%

Table 7: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♟d6+	4.07%	3.57%	4.96%	6.54%	3.94%	2.77%	26.23%	65.34%
♜xa5	50.76%	45.54%	45.91%	51.70%	48.54%	50.24%	41.97%	12.18%
♜xc6+	10.38%	16.40%	25.83%	24.85%	31.97%	28.81%	3.95%	0.61%
♜xd4	19.78%	17.56%	14.54%	9.69%	5.24%	5.33%	0.87%	0.29%
b4	1.03%	3.59%	1.10%	1.88%	4.79%	8.06%	4.34%	0.29%
♟f6+	0.01%	0.02%	0.04%	0.02%	0.01%	0.01%	12.46%	13.67%
♟c5	0.14%	0.19%	0.45%	0.31%	0.25%	0.08%	0.70%	0.29%
f4	0.75%	0.93%	0.66%	0.98%	2.89%	1.73%	1.54%	0.26%
a3	0.65%	0.59%	0.21%	0.32%	0.14%	0.13%	1.41%	0.28%
♟f3	2.91%	2.80%	1.57%	1.31%	0.60%	0.17%	0.86%	0.28%
g4	0.01%	0.01%	0.01%	0.02%	0.02%	0.02%	0.13%	0.29%
♜b4	1.85%	1.26%	0.58%	0.17%	0.11%	0.06%	0.06%	0.27%
♟f3	1.33%	0.75%	0.66%	0.25%	0.09%	0.44%	0.38%	0.29%
♜b3	3.44%	3.32%	1.26%	0.54%	0.18%	0.46%	0.28%	0.29%
h3	0.04%	0.03%	0.02%	0.02%	0.02%	0.02%	0.25%	0.30%
♟g5	0.06%	0.17%	0.01%	0.02%	0.01%	0.10%	0.26%	0.27%
♜a3	0.90%	1.53%	0.67%	0.28%	0.19%	0.13%	0.10%	0.29%
e3	0.25%	0.23%	0.12%	0.09%	0.10%	0.05%	0.43%	0.28%
♟f1	0.04%	0.02%	0.03%	0.02%	0.02%	0.03%	0.13%	0.34%
♟c3	0.08%	0.08%	0.06%	0.05%	0.03%	0.09%	0.72%	0.29%
♟f1	0.07%	0.05%	0.07%	0.07%	0.02%	0.02%	0.12%	0.33%
h4	0.14%	0.09%	0.04%	0.03%	0.03%	0.02%	0.15%	0.33%
b3	0.12%	0.05%	0.03%	0.03%	0.01%	0.01%	0.14%	0.25%
f3	0.05%	0.07%	0.03%	0.02%	0.01%	0.07%	1.11%	0.26%
♟h3	0.16%	0.18%	0.16%	0.04%	0.04%	0.04%	0.13%	0.32%
♜c4	0.23%	0.40%	0.31%	0.30%	0.32%	0.62%	0.31%	0.28%
♜b1	0.18%	0.14%	0.19%	0.03%	0.03%	0.04%	0.23%	0.31%
♟d1	0.15%	0.11%	0.13%	0.10%	0.13%	0.13%	0.16%	0.32%
♜d1	0.06%	0.03%	0.03%	0.03%	0.02%	0.03%	0.13%	0.32%
♜c2	0.14%	0.08%	0.09%	0.11%	0.14%	0.09%	0.16%	0.29%
♟h3	0.09%	0.09%	0.04%	0.01%	0.03%	0.08%	0.22%	0.29%
♜b5	0.11%	0.14%	0.18%	0.14%	0.08%	0.12%	0.07%	0.29%

F.2 Case Study 2: Early and Consistent Solution Lock-In

This puzzle (Figure 10) features a back-rank mate initiated by a queen sacrifice (PV: 1. ♕×e8+ ♖×e8 2. ♜×e8♯). The winning move ♕×e8+ becomes the top candidate from layer 0 and maintains this position throughout all layers, with only brief competition from ♖×c3 at layer 7 (50.8% vs 46.1%). The early convergence may result from the winning move combining both queen and capture biases observed in initial layers. This monotonic pattern contrasts with the delayed convergence seen in previous cases and represents one example where low entropy and KL divergence are maintained across layers from early stages. Full probabilities are in Table 8.

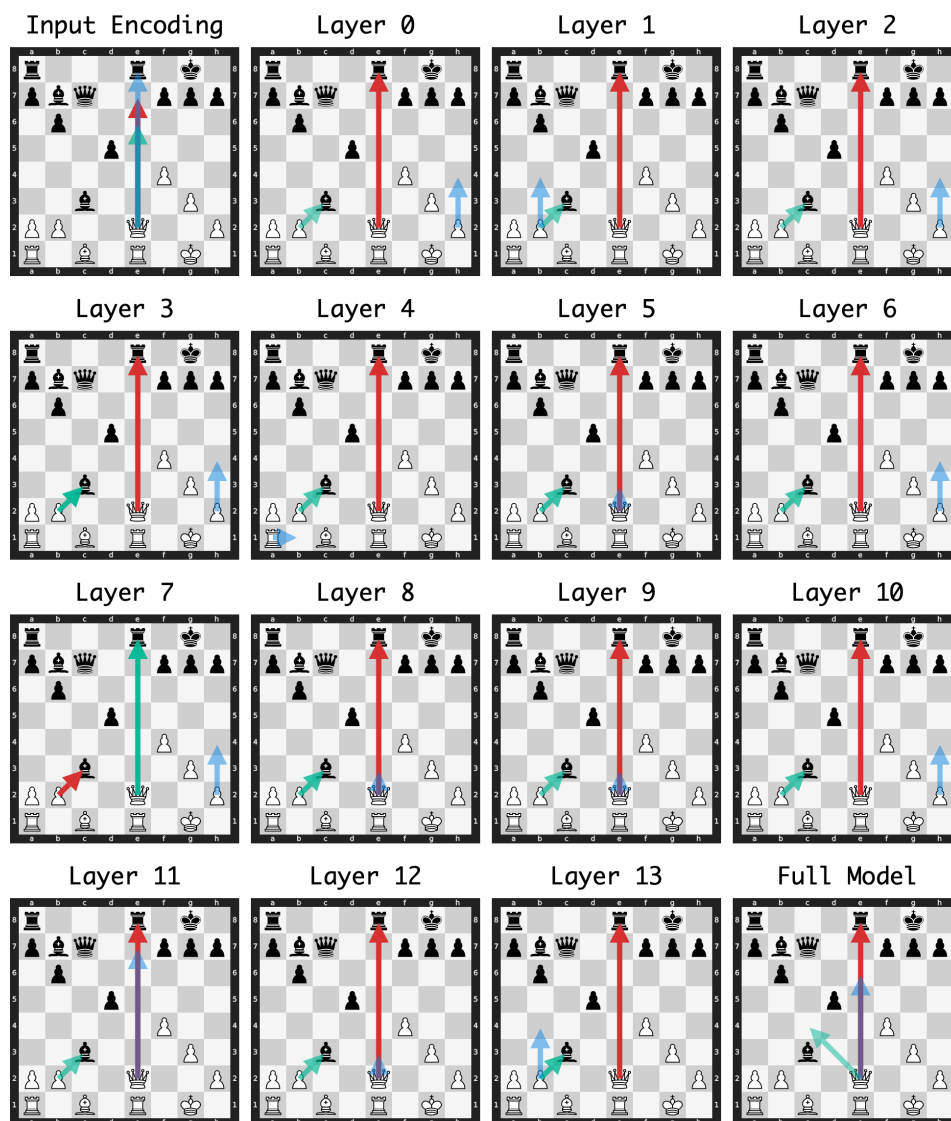


Figure 10: Layer-wise policy evolution for Puzzle 9745.

Table 8: Layer-wise policy probability evolution (Part 1: Input to Layer 6)

Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
♙×e8+	10.74%	51.58%	74.47%	82.24%	51.25%	68.41%	64.05%	66.06%
b×c3	0.55%	13.01%	16.18%	13.09%	45.34%	29.43%	34.47%	30.90%
♙e7	35.40%	0.37%	0.26%	0.07%	0.10%	0.05%	0.03%	0.06%
♙e6	21.65%	1.15%	0.68%	0.09%	0.08%	0.05%	0.06%	0.09%
h4	0.05%	10.18%	2.06%	0.54%	0.52%	0.07%	0.04%	0.26%
b4	1.21%	9.84%	3.20%	0.31%	0.04%	0.02%	0.01%	0.02%
a4	0.05%	6.55%	0.43%	0.09%	0.01%	0.01%	0.04%	0.11%
♙e5	5.75%	1.80%	0.72%	0.49%	0.51%	0.07%	0.05%	0.06%
♙f2	4.09%	0.32%	0.08%	0.18%	0.16%	0.04%	0.05%	0.18%
g4	2.60%	0.74%	0.02%	0.03%	0.07%	0.02%	0.03%	0.09%
♙c4	2.50%	0.11%	0.56%	0.39%	0.12%	0.08%	0.09%	0.15%
♙e4	2.18%	0.03%	0.06%	0.07%	0.07%	0.05%	0.04%	0.12%
♙g2	2.08%	0.02%	0.03%	0.10%	0.04%	0.03%	0.03%	0.06%
♙b5	1.78%	0.02%	0.08%	0.08%	0.05%	0.04%	0.05%	0.07%
♙f3	1.44%	0.04%	0.02%	0.06%	0.03%	0.04%	0.04%	0.05%
a3	0.02%	1.42%	0.07%	0.04%	0.03%	0.04%	0.02%	0.25%
♘h1	1.23%	0.04%	0.06%	0.06%	0.05%	0.06%	0.03%	0.07%
♙g4	1.11%	0.07%	0.07%	0.11%	0.09%	0.07%	0.05%	0.10%
f5	1.07%	0.93%	0.01%	0.01%	0.01%	0.01%	0.02%	0.08%
♙e3	1.00%	0.03%	0.04%	0.04%	0.06%	0.04%	0.18%	0.21%
♘f2	0.40%	0.28%	0.14%	0.51%	0.47%	0.13%	0.05%	0.09%
♙d1	0.15%	0.45%	0.15%	0.15%	0.05%	0.08%	0.03%	0.06%
♙e3	0.13%	0.00%	0.01%	0.06%	0.02%	0.03%	0.08%	0.14%
b3	0.17%	0.29%	0.05%	0.05%	0.04%	0.03%	0.03%	0.11%
♙b1	0.14%	0.02%	0.03%	0.09%	0.08%	0.39%	0.03%	0.08%
♙a6	0.12%	0.07%	0.07%	0.05%	0.10%	0.04%	0.03%	0.04%
♙d3	0.38%	0.02%	0.04%	0.10%	0.06%	0.05%	0.05%	0.07%
♙h5	0.15%	0.07%	0.05%	0.08%	0.10%	0.06%	0.06%	0.09%
♙c2	0.33%	0.03%	0.03%	0.06%	0.04%	0.07%	0.04%	0.08%
♙f1	0.34%	0.02%	0.03%	0.11%	0.03%	0.05%	0.01%	0.01%
h3	0.02%	0.26%	0.06%	0.06%	0.04%	0.01%	0.01%	0.04%
♘g2	0.18%	0.01%	0.04%	0.25%	0.13%	0.11%	0.04%	0.07%
♙d1	0.21%	0.15%	0.10%	0.04%	0.03%	0.16%	0.03%	0.04%
♙d2	0.14%	0.02%	0.03%	0.04%	0.07%	0.04%	0.04%	0.03%
♘f1	0.20%	0.02%	0.03%	0.13%	0.07%	0.07%	0.02%	0.01%
♙f1	0.22%	0.03%	0.03%	0.09%	0.04%	0.03%	0.02%	0.02%
♙d2	0.21%	0.00%	0.01%	0.01%	0.01%	0.02%	0.03%	0.04%

Table 9: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♙×e8+	46.12%	59.87%	71.83%	66.69%	76.07%	73.66%	56.40%	88.91%
♜×c3	50.83%	37.08%	25.57%	29.89%	20.90%	23.58%	34.74%	0.28%
♙e7	0.09%	0.13%	0.13%	0.38%	0.52%	0.18%	0.21%	0.33%
♙e6	0.07%	0.03%	0.03%	0.03%	0.08%	0.07%	0.23%	0.41%
h4	0.37%	0.20%	0.17%	0.53%	0.17%	0.07%	0.24%	0.34%
b4	0.02%	0.04%	0.13%	0.17%	0.06%	0.15%	0.94%	0.33%
a4	0.10%	0.05%	0.02%	0.06%	0.07%	0.10%	0.83%	0.29%
♙e5	0.07%	0.06%	0.07%	0.06%	0.16%	0.18%	0.18%	0.37%
♙f2	0.06%	0.05%	0.03%	0.03%	0.02%	0.04%	0.12%	0.24%
g4	0.10%	0.09%	0.15%	0.25%	0.14%	0.12%	0.58%	0.30%
♙c4	0.12%	0.06%	0.09%	0.07%	0.07%	0.07%	0.32%	0.45%
♙e4	0.09%	0.05%	0.07%	0.08%	0.17%	0.15%	0.18%	0.40%
♙g2	0.05%	0.03%	0.02%	0.02%	0.03%	0.06%	0.13%	0.27%
♙b5	0.06%	0.10%	0.07%	0.05%	0.06%	0.07%	0.17%	0.21%
♙f3	0.06%	0.03%	0.04%	0.03%	0.05%	0.04%	0.15%	0.34%
a3	0.18%	0.09%	0.07%	0.09%	0.05%	0.04%	0.42%	0.32%
♜h1	0.05%	0.04%	0.02%	0.02%	0.03%	0.05%	0.09%	0.33%
♙g4	0.07%	0.04%	0.05%	0.04%	0.05%	0.04%	0.16%	0.35%
f5	0.15%	0.55%	0.22%	0.19%	0.29%	0.14%	0.45%	0.30%
♙e3	0.29%	0.62%	0.26%	0.07%	0.07%	0.20%	0.22%	0.21%
♜f2	0.04%	0.04%	0.04%	0.05%	0.05%	0.09%	0.09%	0.24%
♙d1	0.05%	0.02%	0.04%	0.04%	0.06%	0.04%	0.14%	0.29%
♜e3	0.12%	0.16%	0.20%	0.12%	0.04%	0.08%	0.41%	0.21%
b3	0.12%	0.06%	0.10%	0.07%	0.06%	0.04%	0.30%	0.41%
♜b1	0.05%	0.03%	0.08%	0.39%	0.18%	0.12%	0.40%	0.37%
♙a6	0.05%	0.05%	0.04%	0.04%	0.05%	0.04%	0.14%	0.38%
♙d3	0.07%	0.07%	0.06%	0.03%	0.06%	0.08%	0.17%	0.28%
♙h5	0.07%	0.05%	0.04%	0.06%	0.06%	0.04%	0.24%	0.37%
♙c2	0.11%	0.05%	0.06%	0.06%	0.07%	0.06%	0.16%	0.34%
♜f1	0.01%	0.01%	0.01%	0.01%	0.01%	0.03%	0.17%	0.25%
h3	0.09%	0.04%	0.04%	0.09%	0.04%	0.02%	0.22%	0.33%
♜g2	0.04%	0.04%	0.03%	0.05%	0.09%	0.12%	0.10%	0.27%
♜d1	0.04%	0.03%	0.04%	0.03%	0.03%	0.04%	0.20%	0.27%
♙d2	0.03%	0.03%	0.04%	0.02%	0.01%	0.03%	0.07%	0.27%
♜f1	0.01%	0.01%	0.02%	0.03%	0.04%	0.06%	0.08%	0.26%
♙f1	0.03%	0.02%	0.02%	0.02%	0.05%	0.06%	0.14%	0.26%
♜d2	0.09%	0.07%	0.10%	0.11%	0.05%	0.06%	0.22%	0.23%

F.3 Case Study 3: Knight Sacrifice with Rook Mate

This puzzle (Figure 11) featured in Jenner et al. (2024) requires a knight sacrifice (PV: 1. ♖g6+ ♗h4#). The input layer exhibits piece-specific bias toward knight moves. The winning move ♖g6+ peaks as the top candidate at layer 3, then disappears as the model favors ♖e6 (layers 4-7) and ♗e6 (layers 7-9), the latter threatening the opposing queen while protected by the knight. The correct sacrifice re-emerges from layer 8, becoming the preferred move after layer 10. This pattern demonstrates the model’s exploration of tactically sound alternatives before converging on the optimal sequence. Full probabilities are in Table 10.

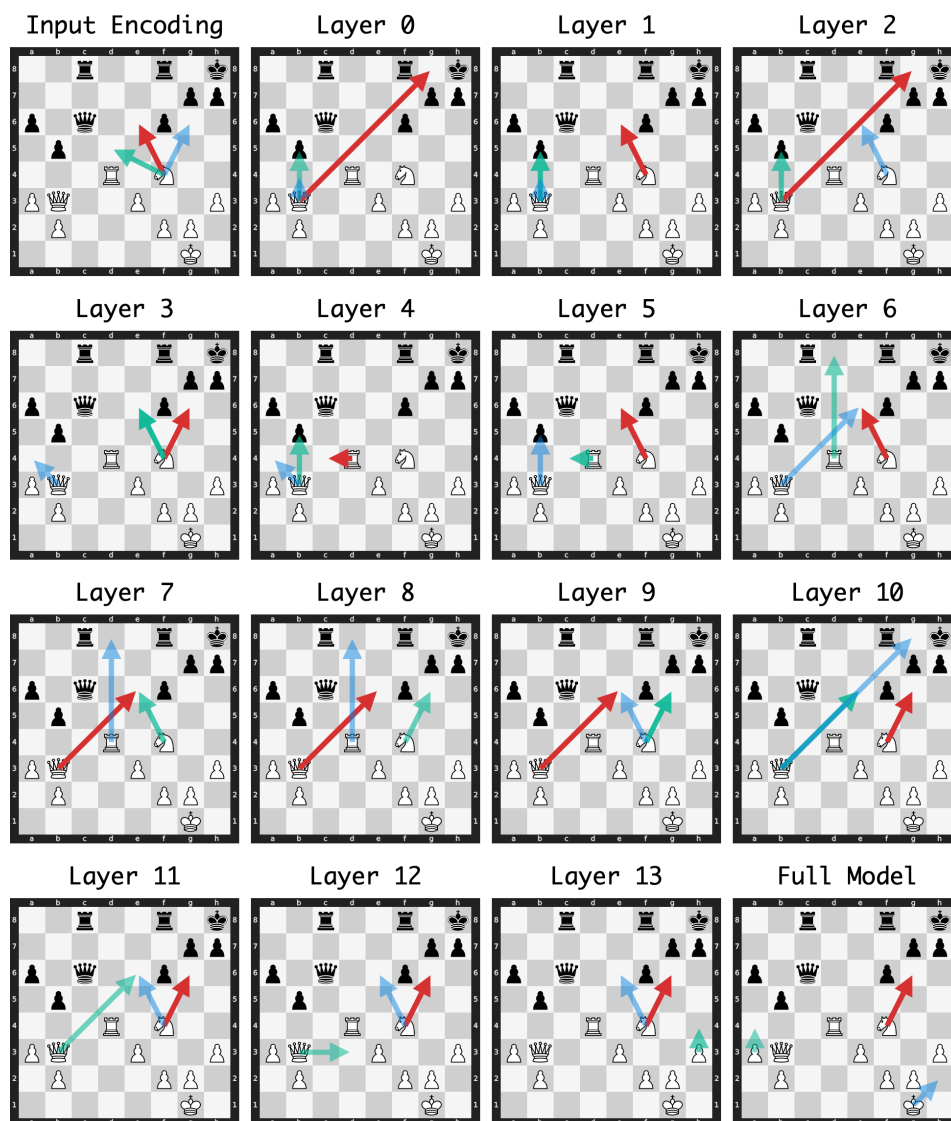


Figure 11: Layer-wise policy evolution for Puzzle 483.

Table 10: Layer-wise policy probability evolution (Part 1: Input to Layer 6)


































Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
 g6+	11.14%	0.37%	0.64%	0.19%	14.28%	0.64%	6.00%	3.24%
 c4	0.87%	0.19%	0.39%	0.77%	6.80%	48.97%	18.55%	9.85%
 g8+	0.04%	37.56%	9.18%	29.26%	8.63%	1.70%	1.71%	2.12%
 e6	20.13%	3.37%	20.25%	9.54%	13.41%	2.89%	26.73%	23.12%
 e6	0.75%	4.64%	2.37%	0.30%	2.86%	1.23%	6.72%	14.76%
h4	0.07%	7.82%	0.18%	0.47%	0.43%	0.14%	0.34%	0.12%
 xb5	5.37%	16.97%	17.42%	16.55%	8.24%	17.00%	8.75%	3.83%
 d8	2.71%	0.08%	0.07%	0.22%	1.35%	1.12%	0.63%	15.68%
 d5	15.13%	0.07%	0.23%	3.24%	5.28%	1.14%	2.13%	0.74%
 b4	0.05%	14.59%	12.53%	4.31%	3.02%	0.45%	0.32%	0.28%
 h2	0.15%	0.54%	0.15%	1.23%	0.60%	1.00%	2.85%	7.21%
 d3	0.27%	0.11%	0.78%	0.46%	0.61%	0.47%	0.34%	0.77%
 b4	0.10%	3.95%	9.58%	1.28%	1.03%	0.10%	0.03%	0.03%
 d7	9.53%	0.30%	0.25%	0.23%	0.43%	0.56%	0.24%	0.43%
 d6	8.93%	0.14%	0.19%	0.34%	0.67%	0.78%	0.21%	0.27%
 a4	0.03%	1.16%	4.29%	5.99%	8.87%	3.53%	6.25%	2.69%
 d3	0.85%	0.09%	0.08%	1.15%	2.33%	0.51%	3.51%	2.67%
 c2	0.35%	0.22%	0.82%	4.07%	2.35%	0.98%	0.24%	1.14%
 a4	0.12%	0.31%	0.90%	1.27%	2.02%	1.44%	7.49%	2.75%
a4	0.06%	2.74%	0.10%	0.34%	0.33%	0.16%	0.28%	0.13%
 d5	5.56%	0.17%	3.27%	1.48%	0.25%	0.08%	0.14%	0.34%
 f7	0.76%	0.52%	4.44%	1.96%	1.69%	0.51%	0.18%	0.34%
 d1	0.04%	0.14%	0.22%	0.38%	0.67%	0.95%	0.28%	0.59%
e4	3.87%	0.39%	1.09%	3.23%	0.82%	0.15%	0.08%	0.14%
 e2	2.93%	0.06%	0.42%	0.09%	0.05%	0.11%	0.22%	0.11%
g4	2.93%	0.39%	0.17%	0.75%	0.87%	0.23%	0.31%	0.23%
 c4	0.87%	0.58%	2.68%	2.76%	2.12%	1.16%	1.68%	1.66%
 d2	0.21%	0.10%	0.15%	0.31%	0.68%	2.37%	0.31%	0.16%
 h1	1.61%	0.31%	0.68%	1.08%	0.30%	2.28%	0.66%	0.47%
 f1	0.26%	0.18%	0.07%	0.42%	1.18%	1.99%	0.28%	0.36%
f3	0.66%	0.09%	0.31%	0.61%	1.97%	0.25%	0.11%	0.07%
 e4	1.21%	0.15%	1.07%	1.43%	1.29%	0.67%	0.49%	1.73%
 d3	0.30%	0.10%	1.65%	0.88%	0.18%	0.09%	0.16%	0.06%
 d5	0.62%	0.13%	1.45%	0.54%	0.76%	0.56%	0.45%	0.62%
 a2	0.02%	0.57%	1.14%	1.06%	1.43%	1.42%	0.56%	0.49%
 c3	0.14%	0.18%	0.36%	1.03%	0.85%	1.20%	0.27%	0.28%
g3	0.45%	0.41%	0.17%	0.38%	1.00%	0.18%	0.14%	0.23%
 h5	0.79%	0.22%	0.04%	0.11%	0.18%	0.39%	0.14%	0.07%
 d1	0.11%	0.08%	0.20%	0.28%	0.19%	0.58%	0.25%	0.22%

Table 11: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♟g6+	8.80%	15.41%	17.79%	22.84%	38.02%	60.31%	34.33%	70.65%
♜c4	5.83%	3.22%	2.65%	1.14%	0.42%	0.13%	0.23%	0.18%
♞g8+	3.09%	2.18%	7.24%	9.71%	1.63%	0.72%	0.53%	0.16%
♞e6	14.17%	13.05%	11.89%	9.52%	9.15%	7.89%	11.68%	1.46%
♞e6	20.13%	19.05%	19.79%	18.19%	12.49%	3.40%	3.67%	0.48%
h4	0.08%	0.19%	0.07%	0.14%	2.00%	2.11%	17.94%	4.76%
♞xb5	4.14%	5.70%	3.74%	2.16%	0.52%	0.58%	1.43%	0.16%
♜d8	10.70%	14.67%	6.79%	4.26%	6.70%	1.06%	0.38%	0.17%
♞d5	0.60%	0.59%	0.40%	0.20%	0.10%	0.10%	0.54%	0.15%
♞b4	0.46%	0.74%	0.55%	0.57%	0.42%	0.55%	0.58%	0.71%
♞h2	10.63%	5.59%	4.27%	3.32%	1.08%	3.43%	3.55%	5.54%
♞d3	2.55%	2.07%	3.87%	5.28%	7.64%	9.98%	6.51%	4.04%
♜b4	0.02%	0.02%	0.01%	0.02%	0.04%	0.05%	0.04%	0.15%
♜d7	0.34%	0.30%	0.34%	0.42%	0.88%	0.38%	0.16%	0.21%
♜d6	0.16%	0.16%	0.15%	0.20%	0.23%	0.13%	0.09%	0.18%
♞a4	1.47%	1.62%	0.72%	1.28%	1.32%	0.33%	0.33%	0.14%
♞d3	3.70%	3.23%	4.31%	2.44%	3.75%	3.09%	8.82%	0.49%
♞c2	1.90%	3.52%	5.13%	7.69%	2.28%	0.20%	0.19%	0.16%
♜a4	1.44%	1.05%	0.10%	0.41%	0.49%	0.13%	0.11%	0.15%
a4	0.25%	0.08%	0.03%	0.04%	0.14%	0.05%	1.32%	5.80%
♜d5	0.27%	0.23%	0.11%	0.08%	0.16%	0.19%	0.34%	0.22%
♞f7	0.90%	1.19%	1.59%	3.37%	2.46%	0.28%	0.26%	0.20%
♞d1	2.18%	1.44%	3.39%	2.26%	3.90%	1.11%	0.65%	0.46%
e4	0.14%	0.10%	0.16%	0.16%	0.15%	0.19%	0.44%	0.18%
♞e2	0.19%	0.12%	0.40%	0.29%	0.50%	0.54%	1.50%	0.18%
g4	0.27%	0.18%	0.08%	0.06%	0.09%	0.06%	0.21%	0.40%
♞c4	1.13%	0.79%	0.65%	0.45%	0.19%	0.13%	0.17%	0.17%
♜d2	0.07%	0.07%	0.11%	0.02%	0.02%	0.06%	0.27%	0.25%
♞h1	0.45%	0.48%	0.63%	0.90%	0.23%	0.27%	0.38%	0.19%
♞f1	0.57%	0.49%	0.49%	0.32%	1.00%	0.95%	0.23%	0.15%
f3	0.18%	0.08%	0.08%	0.06%	0.06%	0.05%	0.40%	0.18%
♜e4	1.11%	0.58%	0.85%	0.77%	0.78%	0.65%	0.29%	0.19%
♜d3	0.06%	0.05%	0.04%	0.05%	0.10%	0.10%	0.23%	0.19%
♞d5	0.76%	0.98%	0.85%	0.76%	0.66%	0.30%	0.70%	0.16%
♞a2	0.61%	0.33%	0.29%	0.19%	0.13%	0.07%	0.33%	0.27%
♞c3	0.25%	0.16%	0.18%	0.26%	0.15%	0.15%	0.14%	0.15%
g3	0.11%	0.06%	0.04%	0.04%	0.01%	0.03%	0.20%	0.29%
♞h5	0.10%	0.09%	0.02%	0.05%	0.05%	0.08%	0.33%	0.18%
♜d1	0.17%	0.13%	0.19%	0.05%	0.08%	0.17%	0.52%	0.25%

F.4 Case Study 4: Queen Sacrifice for Rook Mate

This puzzle (Figure 12) involves a queen sacrifice to enable a rook mate (PV: 1. ♔e8+ ♖×e8 2. ♖×e8♯). The winning move ♔e8+ receives minimal probability until late layers, then increases sharply to become the top choice at layer 13 and dominates the final output (58.9%). Throughout most of the middle to late layers, the model favors g×f7+, which delivers check but lacks the forced mate continuation. Early layers prefer immediate material captures (♗×a3, ♖×a3) that maintain substantial probability until late layers. Full probabilities are in Table 12.

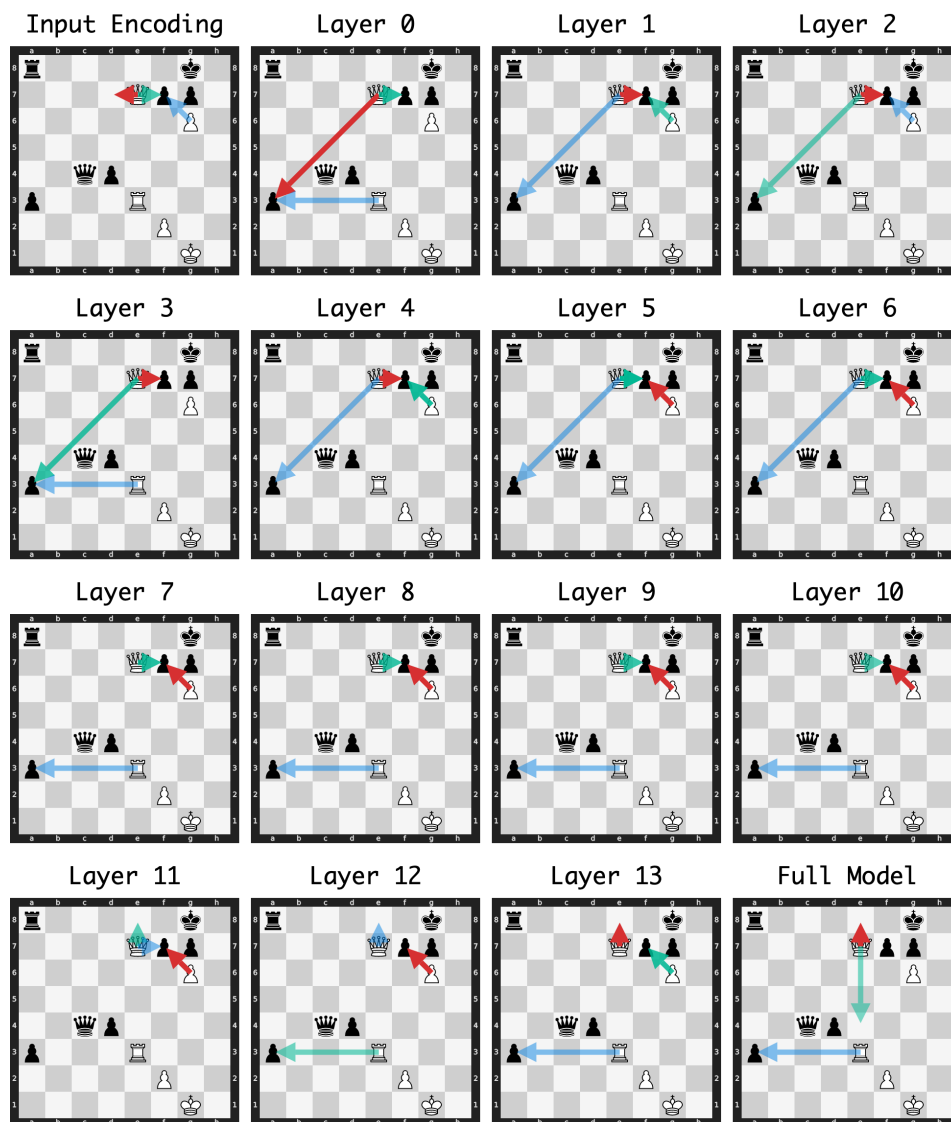


Figure 12: Layer-wise policy evolution for Puzzle 215.

Table 12: Layer-wise policy probability evolution (Part 1: Input to Layer 6)

Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
e8+	2.73%	0.18%	0.30%	0.67%	3.10%	0.03%	0.15%	3.48%
xf7+	13.47%	30.73%	38.98%	32.49%	33.46%	37.33%	33.63%	31.74%
xf7+	12.07%	3.16%	26.81%	27.44%	14.66%	34.00%	33.96%	38.81%
xa3	0.62%	34.10%	18.65%	28.12%	26.36%	15.37%	15.63%	10.99%
xa3	0.05%	22.90%	8.54%	7.36%	16.37%	12.10%	13.80%	10.10%
d7	15.33%	0.07%	0.24%	0.03%	0.03%	0.02%	0.02%	0.01%
c7	11.27%	0.04%	0.04%	0.03%	0.02%	0.02%	0.01%	0.01%
e4	0.23%	0.02%	0.03%	0.04%	0.05%	0.03%	0.06%	0.05%
f6	7.92%	0.04%	0.24%	0.06%	0.06%	0.02%	0.05%	0.03%
g5	2.88%	7.28%	1.71%	0.11%	0.09%	0.04%	0.04%	0.02%
h3	0.01%	0.04%	0.05%	0.26%	0.34%	0.03%	0.70%	1.98%
f3	0.07%	0.04%	0.41%	0.31%	0.14%	0.02%	0.33%	0.50%
b7	2.30%	0.06%	0.05%	0.02%	0.02%	0.02%	0.02%	0.02%
d6	6.25%	0.03%	0.23%	0.03%	0.02%	0.02%	0.02%	0.01%
e6	5.98%	0.05%	0.08%	0.08%	0.04%	0.02%	0.02%	0.02%
c5	4.04%	0.03%	0.02%	0.09%	0.04%	0.02%	0.02%	0.01%
f8+	2.91%	0.14%	1.61%	0.87%	3.51%	0.24%	0.33%	1.00%
e5	3.49%	0.03%	0.09%	0.08%	0.03%	0.02%	0.02%	0.01%
e4	0.13%	0.03%	0.03%	0.05%	0.15%	0.02%	0.17%	0.18%
d8+	2.73%	0.17%	0.43%	0.12%	0.24%	0.02%	0.03%	0.11%
b4	1.54%	0.07%	0.10%	0.21%	0.09%	0.03%	0.01%	0.01%
f4	1.44%	0.10%	0.04%	0.04%	0.03%	0.04%	0.02%	0.03%
g3	0.03%	0.03%	0.10%	0.17%	0.12%	0.02%	0.33%	0.30%
a7	0.70%	0.05%	0.07%	0.04%	0.02%	0.02%	0.01%	0.01%
h4	0.68%	0.11%	0.10%	0.18%	0.04%	0.02%	0.02%	0.03%
g2	0.02%	0.03%	0.43%	0.65%	0.52%	0.12%	0.25%	0.22%
c3	0.03%	0.03%	0.03%	0.02%	0.06%	0.07%	0.02%	0.02%
e6	0.34%	0.06%	0.07%	0.04%	0.05%	0.02%	0.02%	0.02%
e5	0.27%	0.03%	0.16%	0.06%	0.04%	0.02%	0.05%	0.04%
h1	0.23%	0.05%	0.08%	0.05%	0.04%	0.03%	0.06%	0.04%
h2	0.02%	0.04%	0.09%	0.14%	0.12%	0.06%	0.12%	0.13%
b3	0.03%	0.05%	0.05%	0.05%	0.06%	0.05%	0.03%	0.02%
d3	0.05%	0.03%	0.05%	0.05%	0.03%	0.03%	0.03%	0.03%
e2	0.07%	0.03%	0.05%	0.02%	0.02%	0.02%	0.01%	0.01%
e1	0.01%	0.12%	0.04%	0.02%	0.01%	0.05%	0.01%	0.01%
f3	0.09%	0.02%	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%

Table 13: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♙e8+	1.73%	7.55%	7.69%	10.74%	18.80%	17.46%	21.69%	58.86%
♙xf7+	31.45%	26.33%	27.12%	20.10%	16.39%	6.96%	2.25%	0.27%
gxf7+	38.73%	34.13%	38.96%	32.23%	25.25%	34.38%	19.50%	1.46%
♙xa3	11.94%	12.60%	9.55%	8.78%	5.44%	3.29%	7.27%	0.45%
♜xa3	12.33%	13.88%	11.84%	15.80%	16.38%	19.25%	12.83%	9.17%
♙d7	0.01%	0.01%	0.01%	0.01%	0.03%	0.02%	0.18%	0.24%
♙c7	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.30%	0.21%
♙e4	0.06%	0.15%	0.39%	0.57%	0.91%	1.17%	8.04%	9.86%
♙f6	0.04%	0.02%	0.01%	0.02%	0.02%	0.02%	0.13%	0.18%
♙g5	0.02%	0.02%	0.01%	0.01%	0.01%	0.01%	0.06%	0.20%
♜h3	1.36%	2.72%	1.67%	4.23%	6.57%	6.83%	6.64%	6.35%
♜f3	0.35%	0.46%	0.27%	2.00%	5.91%	6.43%	5.91%	1.98%
♙b7	0.01%	0.02%	0.02%	0.03%	0.07%	0.31%	6.30%	4.20%
♙d6	0.01%	0.01%	0.01%	0.01%	0.01%	0.00%	0.06%	0.18%
♙e6	0.02%	0.04%	0.10%	0.04%	0.01%	0.02%	0.06%	0.18%
♙c5	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.06%	0.22%
♙f8+	0.75%	0.60%	0.97%	3.51%	0.60%	0.04%	0.46%	0.22%
♙e5	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.07%	0.18%
♜e4	0.10%	0.18%	0.10%	0.41%	2.02%	2.08%	3.10%	0.75%
♙d8+	0.06%	0.16%	0.13%	0.24%	0.40%	0.01%	0.07%	0.25%
♙b4	0.01%	0.01%	0.02%	0.02%	0.01%	0.01%	0.09%	0.17%
f4	0.06%	0.12%	0.17%	0.27%	0.16%	0.24%	0.93%	0.16%
♜g3	0.22%	0.30%	0.14%	0.35%	0.42%	0.98%	1.22%	0.54%
♙a7	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	1.06%	0.77%
♙h4	0.04%	0.10%	0.11%	0.11%	0.09%	0.07%	0.48%	0.71%
♘g2	0.29%	0.19%	0.27%	0.15%	0.05%	0.01%	0.09%	0.18%
♜c3	0.06%	0.03%	0.04%	0.02%	0.05%	0.05%	0.12%	0.35%
♜e6	0.02%	0.03%	0.03%	0.03%	0.04%	0.03%	0.08%	0.24%
♜e5	0.03%	0.04%	0.02%	0.04%	0.13%	0.14%	0.16%	0.21%
♘h1	0.03%	0.03%	0.03%	0.03%	0.04%	0.05%	0.11%	0.21%
♘h2	0.11%	0.14%	0.12%	0.10%	0.03%	0.01%	0.12%	0.19%
♜b3	0.03%	0.02%	0.03%	0.03%	0.04%	0.04%	0.18%	0.19%
♜d3	0.05%	0.06%	0.08%	0.10%	0.07%	0.06%	0.14%	0.18%
♜e2	0.02%	0.01%	0.02%	0.01%	0.01%	0.01%	0.08%	0.17%
♜e1	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.16%	0.17%
f3	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.01%	0.15%

E.5 Case Study 5: Bishop Sacrifice for Discovered Attack

This puzzle (Figure 13) features a bishop sacrifice that wins material via a discovered attack (PV: 1. ♖h2+ ♗xh2 2. ♜xc6). The model initially favors the immediate material gain ♜xc6 through layer 11, while the winning move ♖h2+ gains probability gradually from layer 5 and becomes the top choice after layer 12, demonstrating a late-stage override of a simple material-gain heuristic in favor of a deeper sacrificial sequence. Full probabilities are in Table 14.

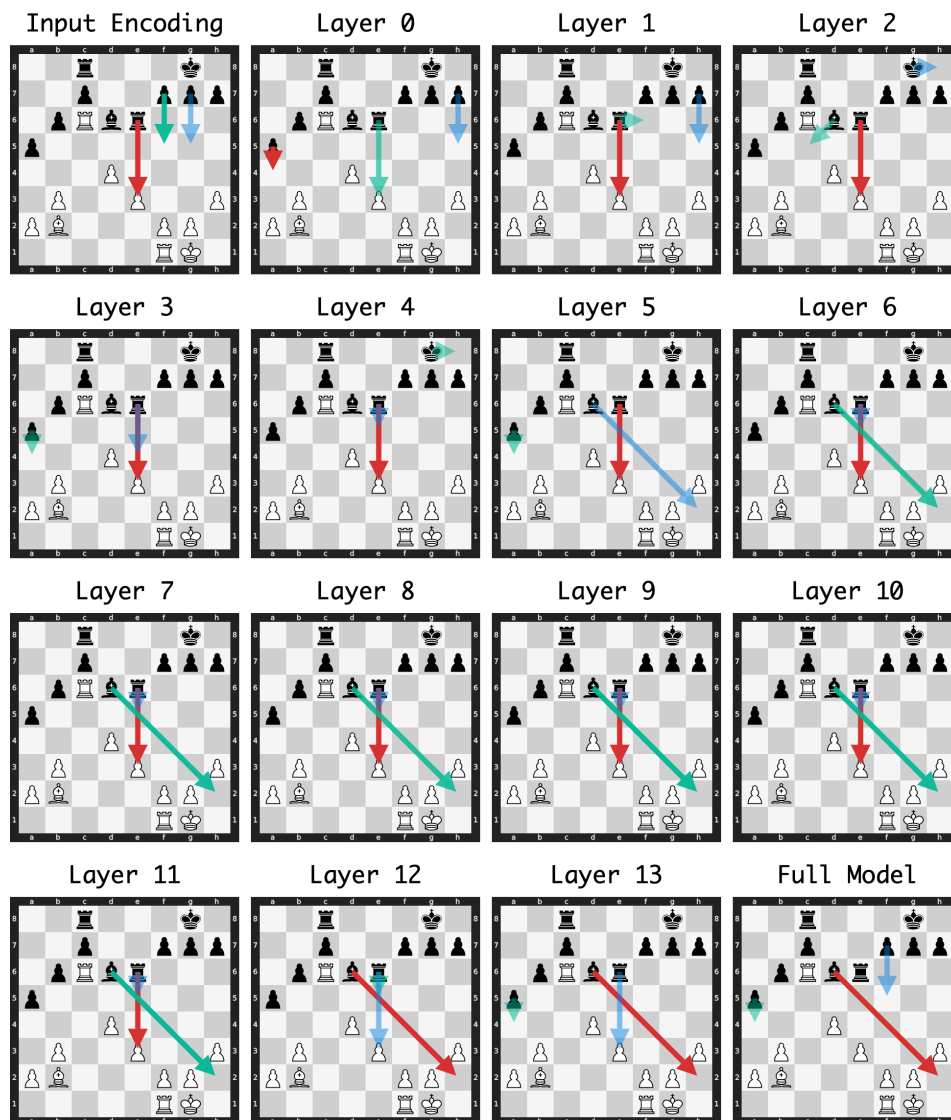


Figure 13: Layer-wise policy evolution for Puzzle 10363.

Table 14: Layer-wise policy probability evolution (Part 1: Input to Layer 6)

Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
♙h2+	0.06%	0.01%	0.65%	0.23%	0.45%	0.21%	3.92%	31.39%
♜x3	25.43%	24.14%	74.33%	83.14%	79.68%	92.11%	74.68%	46.36%
a4	5.81%	47.36%	0.77%	0.21%	9.00%	0.04%	12.92%	8.74%
f5	22.75%	0.42%	0.11%	0.02%	0.05%	0.01%	0.01%	0.17%
♞e5	2.90%	0.01%	0.05%	0.11%	0.23%	1.07%	2.55%	9.16%
h5	0.15%	15.19%	3.82%	0.17%	0.03%	0.00%	0.01%	0.01%
b5	4.88%	10.36%	0.74%	0.54%	0.02%	0.01%	0.05%	0.05%
g5	6.84%	0.35%	0.19%	0.04%	0.01%	0.01%	0.02%	0.03%
♙c5	5.00%	0.01%	0.72%	3.85%	0.36%	0.67%	0.15%	0.13%
♞f6	1.68%	0.02%	4.36%	0.39%	0.27%	0.23%	0.14%	0.39%
♙e5	4.28%	0.01%	0.11%	0.68%	0.27%	0.43%	0.17%	0.29%
♞e4	3.88%	0.01%	3.38%	1.21%	2.38%	0.17%	0.14%	0.15%
♙h8	3.46%	0.06%	0.84%	2.50%	1.36%	1.45%	0.17%	0.11%
♞a8	2.22%	0.05%	0.15%	0.61%	0.14%	0.17%	0.13%	0.09%
♞g6	0.75%	0.01%	2.12%	0.48%	0.60%	0.13%	0.10%	0.05%
♞ee8	0.23%	0.32%	1.94%	1.69%	0.47%	0.08%	0.32%	0.14%
♞ce8	0.68%	0.42%	1.92%	0.79%	0.80%	0.10%	0.63%	0.20%
f6	0.69%	0.10%	0.03%	0.07%	0.00%	0.01%	0.01%	0.14%
♙a3	0.30%	0.01%	0.21%	0.13%	1.31%	0.31%	1.46%	0.76%
♙e7	1.44%	0.01%	0.08%	0.15%	0.19%	0.13%	0.16%	0.05%
♞h6	0.11%	0.01%	1.21%	0.36%	0.42%	0.31%	0.17%	0.22%
h6	0.06%	0.85%	0.16%	0.09%	0.01%	0.02%	0.02%	0.05%
♙g3	0.57%	0.02%	0.32%	0.22%	0.24%	0.29%	0.11%	0.13%
♞e7	1.04%	0.01%	0.62%	0.15%	0.16%	0.16%	0.12%	0.07%
♞b8	0.92%	0.02%	0.07%	0.32%	0.11%	0.30%	0.11%	0.11%
♙b4	0.25%	0.01%	0.18%	0.39%	0.46%	0.25%	0.91%	0.33%
♞f8	0.89%	0.02%	0.27%	0.21%	0.18%	0.12%	0.10%	0.04%
g6	0.88%	0.15%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%
♙f4	0.66%	0.01%	0.15%	0.41%	0.18%	0.36%	0.22%	0.28%
♞d8	0.62%	0.02%	0.18%	0.31%	0.13%	0.18%	0.12%	0.07%
♙f8	0.55%	0.02%	0.23%	0.37%	0.36%	0.53%	0.23%	0.16%
♙f8	0.01%	0.01%	0.08%	0.15%	0.15%	0.13%	0.16%	0.11%

Table 15: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♟h2+	36.07%	36.91%	41.18%	39.83%	35.44%	67.90%	68.57%	92.87%
♚xe3	40.66%	52.04%	44.63%	45.31%	36.86%	6.02%	3.89%	0.17%
a4	3.69%	0.97%	0.22%	0.64%	2.16%	3.45%	7.54%	1.08%
f5	0.71%	1.52%	0.28%	0.20%	0.83%	0.37%	1.00%	0.71%
♚e5	16.34%	5.68%	10.72%	11.30%	19.27%	18.62%	2.00%	0.17%
h5	0.02%	0.02%	0.01%	0.01%	0.03%	0.02%	0.43%	0.26%
b5	0.09%	0.08%	0.04%	0.04%	0.17%	0.02%	0.92%	0.15%
g5	0.04%	0.03%	0.03%	0.02%	0.09%	0.04%	0.49%	0.16%
♟c5	0.09%	0.17%	0.13%	0.11%	0.13%	0.15%	0.68%	0.16%
♚f6	0.18%	0.10%	0.16%	0.10%	0.92%	0.22%	0.28%	0.16%
♟e5	0.13%	0.20%	0.13%	0.09%	0.34%	0.23%	0.53%	0.16%
♚e4	0.06%	0.06%	0.07%	0.06%	0.17%	0.10%	0.64%	0.16%
♟h8	0.08%	0.12%	0.26%	0.42%	0.57%	0.60%	1.33%	0.19%
♚a8	0.07%	0.05%	0.05%	0.05%	0.05%	0.05%	0.30%	0.30%
♚g6	0.02%	0.03%	0.02%	0.01%	0.15%	0.06%	0.48%	0.15%
♚ee8	0.07%	0.06%	0.04%	0.03%	0.22%	0.13%	0.65%	0.21%
♚ce8	0.19%	0.14%	0.22%	0.21%	0.30%	0.22%	0.34%	0.39%
f6	0.09%	0.05%	0.02%	0.03%	0.06%	0.03%	1.66%	0.18%
♟a3	0.45%	0.46%	0.37%	0.25%	0.41%	0.13%	0.96%	0.16%
♟e7	0.04%	0.08%	0.08%	0.09%	0.07%	0.16%	0.44%	0.16%
♚h6	0.08%	0.06%	0.06%	0.04%	0.38%	0.10%	0.42%	0.15%
h6	0.02%	0.00%	0.01%	0.01%	0.06%	0.02%	1.19%	0.14%
♟g3	0.10%	0.09%	0.14%	0.12%	0.27%	0.27%	1.06%	0.19%
♚e7	0.01%	0.04%	0.03%	0.02%	0.09%	0.06%	0.42%	0.17%
♚b8	0.04%	0.03%	0.04%	0.04%	0.05%	0.04%	0.31%	0.17%
♟b4	0.15%	0.27%	0.28%	0.12%	0.11%	0.20%	0.61%	0.15%
♚f8	0.03%	0.05%	0.07%	0.11%	0.15%	0.10%	0.34%	0.17%
g6	0.00%	0.00%	0.00%	0.00%	0.01%	0.00%	0.61%	0.14%
♟f4	0.26%	0.32%	0.44%	0.41%	0.33%	0.40%	0.81%	0.18%
♚d8	0.03%	0.04%	0.06%	0.06%	0.07%	0.07%	0.29%	0.27%
♟f8	0.12%	0.18%	0.10%	0.15%	0.16%	0.13%	0.43%	0.17%
♟f8	0.08%	0.18%	0.12%	0.13%	0.06%	0.07%	0.37%	0.14%

E.6 Case Study 6: Queen Sacrifice to Back Rank Mate

This puzzle (Figure 14) features a queen sacrifice leading to a back-rank mate (PV: 1. ♕e1+ ♖e1 2. ♖e1♯). The winning move ♕e1+ exhibits non-monotonic behavior: receiving consideration in layers 1-3 (peak 17.9% at layer 3), disappearing at layer 4 (0.13%), then increasing to 79.6% in the final output. Competing moves in later layers include material captures ♖xf7 and ♘b2+, with the latter persisting under the top three moves until layer 10 despite leading to immediate recapture. Full probabilities are in Table 16.

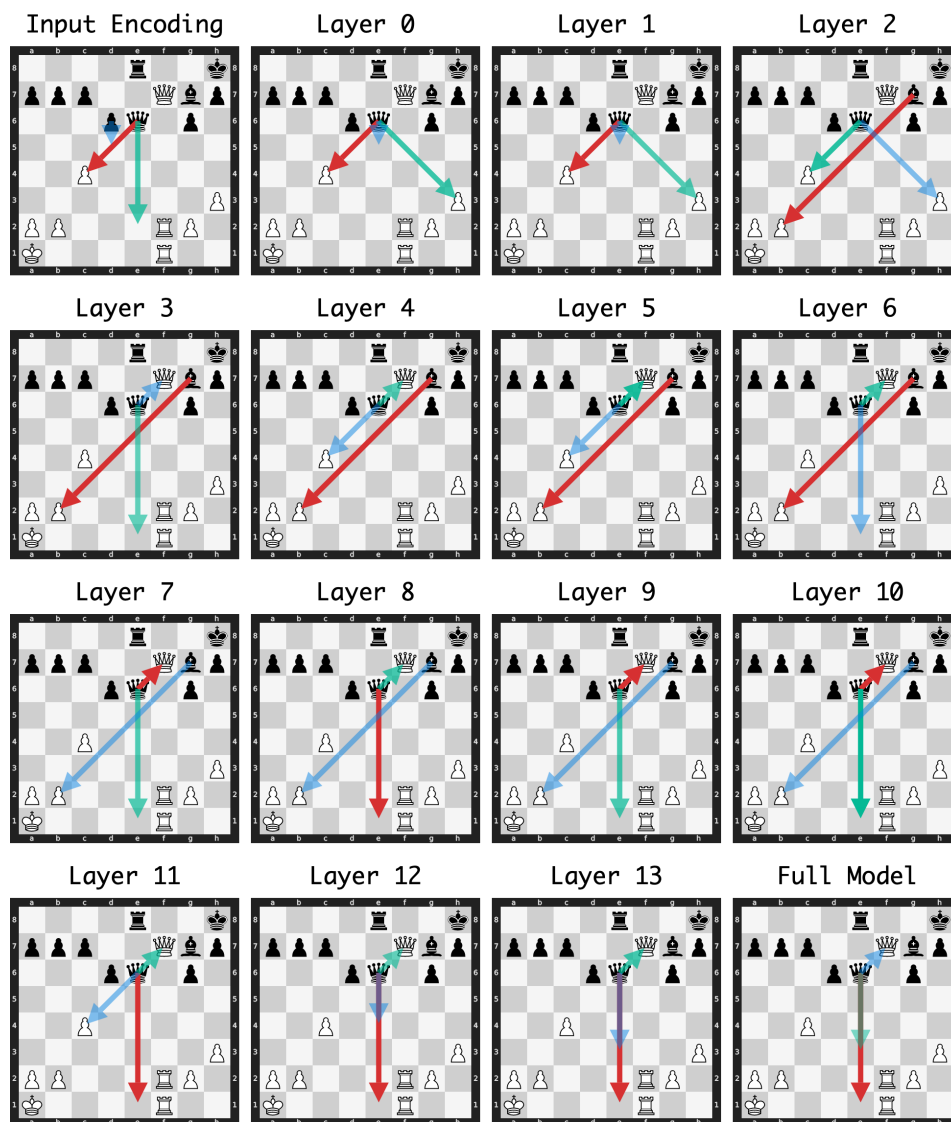


Figure 14: Layer-wise policy evolution for Puzzle 945.

Table 16: Layer-wise policy probability evolution (Part 1: Input to Layer 6)

Move	Input	Layer 0	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
♙e1+	0.59%	0.08%	3.28%	4.64%	17.91%	0.13%	1.31%	24.94%
♜xb2+	5.81%	2.34%	3.76%	23.00%	47.49%	49.87%	37.94%	25.68%
♙xf7	1.86%	10.83%	12.22%	12.21%	9.86%	25.84%	34.15%	25.31%
♙xc4	19.73%	23.07%	25.79%	21.47%	8.31%	14.34%	19.08%	15.96%
♙xh3	3.01%	17.62%	22.63%	17.96%	4.93%	6.23%	3.84%	2.97%
♙e5	0.69%	11.59%	20.98%	2.26%	0.31%	0.12%	1.11%	1.78%
♙e2	13.68%	0.11%	0.08%	0.10%	0.14%	0.04%	0.04%	0.49%
♙e4	4.04%	0.08%	0.14%	0.51%	0.24%	0.09%	0.33%	0.89%
d5	8.31%	11.55%	2.24%	0.90%	0.05%	0.03%	0.03%	0.13%
♙e3	5.69%	0.06%	0.38%	0.43%	0.20%	0.09%	0.19%	0.24%
c5	6.12%	0.65%	0.66%	0.21%	0.02%	0.02%	0.05%	0.03%
♙f5	5.80%	0.08%	0.11%	0.53%	0.40%	0.20%	0.05%	0.04%
g5	5.66%	3.86%	0.04%	0.30%	0.08%	0.01%	0.01%	0.01%
♚g8	0.61%	0.05%	0.21%	5.26%	0.88%	0.06%	0.05%	0.07%
b5	2.70%	4.54%	0.12%	0.16%	0.03%	0.01%	0.02%	0.02%
♙e7	0.93%	0.03%	0.21%	0.29%	0.26%	0.10%	0.06%	0.11%
♙d5	4.03%	0.57%	1.98%	1.52%	1.56%	0.13%	0.12%	0.15%
h5	0.11%	3.69%	0.81%	1.05%	0.29%	0.05%	0.07%	0.09%
a5	0.10%	3.04%	0.04%	0.06%	0.03%	0.01%	0.01%	0.00%
♜e5	0.11%	2.53%	1.90%	0.74%	0.30%	0.10%	0.47%	0.33%
h6	0.05%	1.96%	0.12%	0.31%	0.19%	0.03%	0.04%	0.07%
♚a8	1.67%	0.09%	0.23%	0.34%	0.67%	0.25%	0.07%	0.03%
♜c3	1.24%	0.05%	0.18%	0.50%	0.27%	0.12%	0.05%	0.04%
♙c8	0.09%	0.05%	0.07%	0.24%	0.22%	0.11%	0.04%	0.03%
♜h6	0.02%	0.15%	0.46%	0.51%	1.16%	0.16%	0.08%	0.04%
♙g4	0.75%	0.20%	0.37%	0.87%	0.96%	0.16%	0.11%	0.06%
♜f6	0.86%	0.03%	0.02%	0.21%	0.26%	0.12%	0.03%	0.01%
♚f8	0.68%	0.06%	0.10%	0.84%	0.51%	0.54%	0.10%	0.14%
♙f6	0.73%	0.03%	0.02%	0.22%	0.27%	0.15%	0.04%	0.03%
♚b8	0.70%	0.06%	0.15%	0.24%	0.38%	0.13%	0.04%	0.02%
♜d4	0.69%	0.05%	0.12%	0.26%	0.17%	0.06%	0.08%	0.06%
♙d7	0.62%	0.03%	0.08%	0.33%	0.41%	0.10%	0.02%	0.02%
c6	0.57%	0.06%	0.03%	0.18%	0.14%	0.06%	0.12%	0.06%
♚e7	0.48%	0.04%	0.18%	0.29%	0.25%	0.12%	0.03%	0.04%
♚d8	0.48%	0.05%	0.08%	0.14%	0.12%	0.12%	0.06%	0.04%
♜f8	0.00%	0.05%	0.10%	0.40%	0.48%	0.17%	0.07%	0.04%
a6	0.04%	0.44%	0.03%	0.17%	0.11%	0.02%	0.02%	0.01%
♚c8	0.39%	0.05%	0.07%	0.12%	0.08%	0.06%	0.03%	0.02%
b6	0.35%	0.15%	0.02%	0.23%	0.05%	0.03%	0.04%	0.01%

Table 17: Layer-wise policy probability evolution (Part 2: Layer 7 to Final)

Move	Layer 7	Layer 8	Layer 9	Layer 10	Layer 11	Layer 12	Layer 13	Final
♖e1+	24.64%	35.02%	24.27%	34.15%	40.43%	36.69%	38.62%	79.64%
♜x b2+	21.70%	15.85%	11.51%	9.60%	3.70%	1.89%	2.67%	0.20%
♞xf7	29.86%	24.28%	41.84%	34.92%	21.48%	17.89%	23.77%	4.48%
♞xc4	14.16%	12.55%	11.00%	9.50%	8.00%	5.18%	0.56%	0.10%
♞xh3	2.74%	3.40%	2.12%	1.45%	0.88%	1.46%	2.05%	0.15%
♞e5	1.01%	2.26%	2.81%	1.78%	5.98%	9.57%	4.52%	1.89%
♞e2	1.44%	1.22%	0.23%	1.14%	0.42%	0.14%	0.36%	0.12%
♞e4	1.13%	1.95%	2.52%	2.40%	7.98%	12.92%	6.97%	2.75%
d5	0.12%	0.07%	0.10%	0.07%	0.39%	0.02%	0.30%	0.27%
♞e3	0.29%	0.44%	0.59%	0.72%	4.44%	9.13%	11.24%	5.13%
c5	0.02%	0.03%	0.02%	0.04%	0.02%	0.01%	0.12%	0.13%
♞f5	0.06%	0.04%	0.04%	0.03%	0.03%	0.03%	0.10%	0.14%
g5	0.01%	0.01%	0.01%	0.01%	0.01%	0.02%	0.09%	0.15%
♜g8	0.07%	0.06%	0.04%	0.06%	0.03%	0.02%	0.08%	0.12%
b5	0.06%	0.01%	0.00%	0.01%	0.01%	0.02%	0.56%	0.17%
♞e7	0.28%	0.63%	0.85%	1.73%	4.44%	3.46%	3.03%	0.33%
♞d5	0.13%	0.14%	0.18%	0.16%	0.10%	0.04%	0.12%	0.15%
h5	0.97%	0.80%	0.49%	0.41%	0.46%	0.22%	0.50%	0.30%
a5	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.18%	0.13%
♜e5	0.25%	0.59%	0.37%	0.12%	0.03%	0.04%	0.09%	0.14%
h6	0.16%	0.10%	0.11%	0.15%	0.05%	0.04%	0.37%	0.13%
♜a8	0.04%	0.02%	0.05%	0.12%	0.06%	0.09%	0.16%	0.12%
♜c3	0.03%	0.02%	0.02%	0.02%	0.03%	0.03%	0.11%	0.13%
♞c8	0.09%	0.04%	0.04%	0.03%	0.08%	0.35%	1.20%	1.18%
♜h6	0.02%	0.02%	0.01%	0.01%	0.00%	0.01%	0.08%	0.11%
♞g4	0.10%	0.05%	0.10%	0.16%	0.12%	0.06%	0.14%	0.14%
♜f6	0.01%	0.02%	0.01%	0.01%	0.00%	0.01%	0.06%	0.14%
♜f8	0.14%	0.10%	0.26%	0.55%	0.38%	0.06%	0.18%	0.09%
♞f6	0.04%	0.04%	0.05%	0.04%	0.03%	0.03%	0.06%	0.12%
♜b8	0.03%	0.01%	0.01%	0.00%	0.00%	0.01%	0.08%	0.10%
♜d4	0.04%	0.04%	0.03%	0.02%	0.03%	0.03%	0.33%	0.31%
♞d7	0.04%	0.02%	0.03%	0.03%	0.03%	0.03%	0.13%	0.15%
c6	0.07%	0.03%	0.03%	0.05%	0.02%	0.01%	0.14%	0.11%
♜e7	0.03%	0.04%	0.10%	0.18%	0.24%	0.43%	0.55%	0.12%
♜d8	0.06%	0.02%	0.02%	0.01%	0.01%	0.01%	0.09%	0.11%
♜f8	0.07%	0.05%	0.10%	0.30%	0.06%	0.04%	0.17%	0.10%
a6	0.03%	0.01%	0.00%	0.00%	0.00%	0.00%	0.08%	0.12%
♜c8	0.02%	0.02%	0.03%	0.02%	0.01%	0.02%	0.03%	0.09%
b6	0.02%	0.01%	0.01%	0.01%	0.00%	0.00%	0.12%	0.15%