MODEL SPACE AND FEATURES: ANALYSIS OF INTERNAL REPRESENTATIONS IN LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

How do model resources shape internal representations, and how does their organization relate to performance? We study feature geometry and superposition in decoder-only Transformers trained on random legal move sequences from Othello and Chess, using rule-defined board-state properties as a common reference. Across 128 Dense and Mixture-of-Experts models varying in width, depth, and FFN structure, increasing width primarily separates already represented state directions, whereas order-dependent state properties depend more on depth and FFN capacity. The relationship between summed feature dimensionality and performance differs across tasks: Chess source-square prediction closely tracks this measure, whereas deeper Othello models improve candidate rankings at similar values. Interventions along estimated feature directions causally change the corresponding output probabilities. Excess loss is approximately log-linear in summed feature dimensionality, and the fitted relationship transfers to FFN structures excluded from fitting. These findings characterize how feature geometry and performance vary with model resources, and locate the performance differences that the measured dimensionality does not account for.

1 INTRODUCTION

As Neural Network based Language Models have grown in scale and their architectures have become more complex (Kaplan et al., 2020; Hoffmann et al., 2022), models have acquired wider representational spaces and the ability to perform deeper inference. This, however, also complicates the relationships between the internal states and the computations that humans must interpret. It has consequently become difficult to map a particular internal state directly onto a single concept or behavioral cause, and to fully trace or control the entire process by which that state leads to a final behavior. This difficulty of understanding the mechanism by which internal representations and computations lead to observable behavior is called the black-box property of a model.

To understand models with this property, a human-interpretable approach is to analyze input and output behavior and infer the underlying representations post hoc. However, because a model's output is the final result of combining a vast number of internal representations, it is practically difficult to identify the internal mechanism that produced a given behavior even when the same behavior is observed, and even more so to analyze every individual output and its cause amid the flood of information a model produces. If our understanding of the basis of model behavior fails to keep pace while the range of model applications widens, models gradually slip out of our hands. This constrains not only predicting and controlling model behavior but also harnessing its capabilities, that is, steering them in the desired direction and using them fully for their intended purpose.

To address these limitations, prior work has investigated the internal representations and computations of models through circuit analysis (Olah et al., 2020; Elhage et al., 2021), causal interventions (Vig et al., 2020; Li et al., 2023), feature analysis with sparse autoencoders (SAEs) (Bricken et al., 2023; Cunningham et al., 2023; Gao et al., 2024), and studies of superposition (Elhage et al., 2022; Scherlis et al., 2022). However, decoding particular information from the inside (Ravichander et al., 2021; Elazar et al., 2021; Xu et al., 2020) or observing the geometric relationships among features (Park et al., 2024; Gurnee et al., 2026) does not by itself explain how those representations contribute to actual predictions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This study starts from the view that a model's performance should be explained not by the size of its representational space alone (Allen-Zhu & Li, 2024), but by also examining which features are distinguished in that space and how those distinctions connect to actual choices. Through an analysis of 128 Transformer-based language models trained on Othello and Chess (Li et al., 2023; Nanda et al., 2023; Toshniwal et al., 2022; Karvonen, 2024), varying in width, depth, and FFN structure, we interpret the relationships among model resources, structure, and generation performance from the perspective of Model Space, which treats the geometry of representations together with the computation that transforms them.

Our analysis shows that model resources tend to reach performance through two places. Width separates already represented features from one another and thereby widens the feature arrangement, while depth and the FFN build the arrangement when the state required by the task must be computed in order from the move record. The final blocks compute the ranking of candidates over this arrangement and choose the next move. The range over which performance follows r, the summed feature dimensionality (Elhage et al., 2022) that summarizes the arrangement, differed across tasks. In Chess, where the state must be constructed by tracking move order, performance follows r; in Othello, whose state is fixed once a square is played, performance separates by depth even at the same r. In both games, excess loss decreased log-linearly with the summed feature dimensionality, and most predictions whose evaluation criteria were fixed in advance held. This shows the possibility of conditional empirical laws that connect model conditions to performance through Model Space.

The main contributions of this study are as follows. (i) To investigate the relationships among Information, Size, Sparsity, Feature, and Performance, we define the distribution of internal representations as Model Space, using rule-defined state properties as a common reference (Karvonen et al., 2024). This makes the feature arrangement within it, and the computation that carries it to the output, comparable across different models. (ii) We compare feature arrangements across width, depth, and FFN structure (Dense, and MoE, a structure that selects and computes only some experts (Shazeer et al., 2017; Fedus et al., 2022; Roller et al., 2021)), and trace the process by which the arrangement leads to the output through error composition, candidate rankings, the logit lens, and causal interventions in both games. Through this, we separate the share of performance explained by r from the share it does not explain, and show that the boundary depends on the features the task requires. (iii) We present a log-linear relationship between excess loss and the summed feature dimensionality in both games. We evaluate predictions whose evaluation criteria were fixed in advance on a new width, new seeds, and a different game, report their successes and failures together, and present the grounds and scope on which this relationship can develop into a conditional empirical law.

2 Background

2.1 Board Games: Controlled Information

Board games offer a controlled setting characterized by finite state spaces and explicit rules, naturally bounding the scope of knowledge required for analysis. Within this environment, interpretable features, such as square occupancy, piece ownership, and piece type, are unambiguously defined and can be fully reconstructed from sequential move histories. These features provide a common semantic basis for comparing internal representations across models. Othello and Chess share these properties and have been used to study internal representations and evaluate interpretability methods (Karvonen et al., 2024; Li et al., 2023).

2.2 Representation

Feature. A feature is a property of an input or state captured by a model and may be represented along a particular direction in activation space (Park et al., 2024). Let $f_i(z)$ denote the value of feature i for an input or state z, and let $w_i \in \mathbb{R}^d$ denote its corresponding direction (Elhage et al., 2022). When feature contributions combine linearly, the activation $x(z)$ can be expressed as

$$x(z) = \sum_{i=1}^{N} f_i(z)\, w_i. \tag{1}$$

Here, N is the number of features included in the representation, and d is the dimensionality of the activation space.

Superposition. When the number of represented features N exceeds the activation dimension d, multiple feature directions share the available dimensions. This form of representation is called superposition. Nonorthogonality between feature directions means that projecting an activation onto one direction can also pick up contributions from other features. These cross-contributions constitute interference, whose magnitude depends on the inner products between directions and the feature values.

2.3 SPARSITY

Sparsity refers to different properties depending on the context. Feature sparsity describes how rarely a feature takes a nonzero value across inputs. Greater feature sparsity can favor superposition because fewer features are simultaneously active. Computational sparsity describes the execution of only part of a model for each input. A Mixture-of-Experts (MoE) architecture divides the feed-forward network (FFN) into multiple experts and activates only a subset for each token, separating the total parameter count from the active parameter count used to process a token. In this paper, we define sparsity s as the fraction of the total FFN activated for a token. In MoE models, computational sparsity refers to the selective activation of experts for each token. This is distinct from how features are organized within the representation space (Shazeer et al., 2017; Fedus et al., 2022).

3 MODEL SPACE

Given a defined scope of information, understanding performance differences across width, depth, and sparsity requires examining how features are arranged internally and how those arrangements influence the model's choices. We therefore define **"Model Space"** as the internal space in which a model represents information through features across its width and depth, and study its relationship to predictions through subsequent computation and interventions. In Transformers, we observe this space through residual stream states $x_\ell \in \mathbb{R}^d$ after each block $\ell = 1, \ldots, L$. Because different input histories can produce different states at the same token and position, we characterize Model Space by state distributions conditioned on context c, specified here by the current token and game stage:

$$\mathcal{M}_\theta = \{\mathcal{P}_\theta(x_\ell \mid c)\}_{\ell,c}. \tag{2}$$

Here, θ denotes the learned parameters, width d specifies the dimensionality of the space containing feature directions, and depth L is the number of blocks traversed by ℓ. We call the configuration of feature directions within this space the feature arrangement. The remaining blocks $\ell + 1, \ldots, L$ and the output layer transform the residual state, together with the other prefix states, into a distribution over the next token.

3.1 REPRESENTATION GEOMETRY IN MODEL SPACE

To measure how features are arranged in Model Space, we estimate their directions in the residual stream and quantify how much those directions overlap. This requires a common definition of the information associated with each feature. We describe board-state information through 128 binary indicators of own-piece and opponent-piece occupancy, defined relative to the player to move (Nanda et al., 2023). These properties provide a shared semantic reference across models. At each game stage, we collect residual activations after each block and center both activations and property values conditional on the current token. With centered matrices $H \in \mathbb{R}^{n\times d}$ and $F \in \mathbb{R}^{n\times 128}$, we estimate the directions through joint ridge regression:

$$\widehat{W} = \underset{W}{\operatorname{argmin}} \left\| H - FW^\top \right\|_F^2 + \lambda \left\| W \right\|_F^2 . \tag{3}$$

Each column w_i of $\widehat{W} \in \mathbb{R}^{d\times 128}$ is the estimated direction associated with property i, accounting jointly for the other properties. In the reconstruction $\widehat{x} = \sum_i f_i w_i$, the centered property value f_i

scales that direction's contribution to the activation. We therefore characterize feature arrangement through relationships among these vectors within each model and block. Directions are estimated and evaluated on separate games, with label-shuffling controls and split-sample comparisons used to assess stability. Detection criteria are provided in Appendix A.

Using these estimated directions, we quantify the geometric arrangement of features through feature dimensionality, which reflects how each direction shares the representation space with others (Elhage et al., 2022):

$$D_i = \frac{\|w_i\|^4}{\sum_{j=1}^{128}(w_i^\top w_j)^2}, \qquad r = \sum_{i\in\mathcal{S}} D_i. \tag{4}$$

Here, $\mathcal{S}$ is the set of detected directions, and the denominator includes all 128 estimated directions. The dimensional contribution D_i equals 1 when a direction is orthogonal to all other directions and decreases as their squared inner products increase relative to its own magnitude. The sum r summarizes the dimensional allocation to the selected state directions and is bounded by width d. To measure directional overlap separately, we multiply the mean squared cosine between represented own-piece directions for different squares by d. This overlap ratio has an expectation of 1 for independent isotropic random directions. As a separate check on information decodability, we train linear probes on the full residual activations and report AUC for each property.

3.2 Measuring Performance and Causal Effects

To relate the measured feature arrangements to performance, we examine which candidates the model selects and what kinds of errors it makes. We measure performance by legal-move rate and separate incorrect selections into state errors, which conflict with board occupancy or ownership, and rule errors, which violate move constraints among otherwise eligible candidates. This distinction helps relate performance changes to state distinctions or to the selection of moves under the rules. To examine candidate selection more closely, we compute the AUC for distinguishing legal and illegal candidates using their logits within each board position and average it across positions. We also apply a logit lens, using the final normalization and output layer to read intermediate residual states, to examine where these rankings become available under the output readout.

To connect these selection patterns to the estimated feature directions, we add or subtract a direction from the residual activation and execute the remaining computation. We compare the resulting change in the probability of choosing the target square with the effect of a random direction of matched magnitude, evaluating the causal influence of the selected direction on the output. This intervention complements the observed relationships between representational geometry and error composition by testing whether the estimated directions affect actual choices.

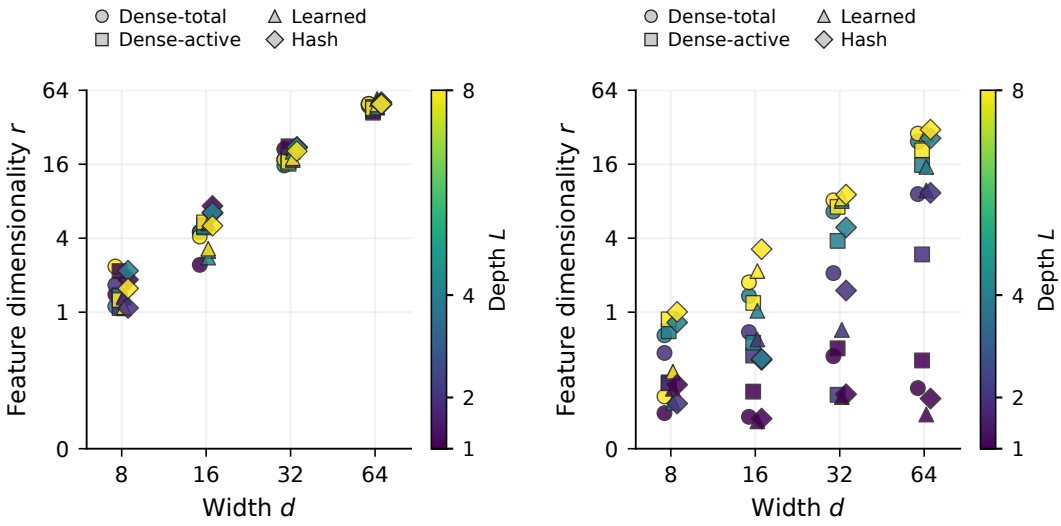
We then quantify how well feature arrangements explain performance differences across models. To account for target uncertainty that varies across games and stages, we use excess loss $\mathcal{E}_t$, defined as cross-entropy minus the entropy of the target distribution. At each stage t, we fit an empirical relationship between excess loss and the summed feature dimensionality r defined above:

$$\mathcal{E}_t \approx a_t - b_t \log r. \tag{5}$$

We fit a baseline using $\log d$ instead of $\log r$ on the same data to assess how well the measured feature arrangement explains performance compared with nominal representational width.

3.3 Models and Data

To examine how width, depth, and FFN structure affect feature arrangements and performance, we vary these three factors while holding the remaining conditions fixed. The design is the same for both games: four FFN structures (Dense-total, Dense-active, Learned, and Hash), widths $d \in \{8, 16, 32, 64\}$, and depths $L \in \{1, 2, 4, 8\}$, one model per cell with seed 0, giving 64 models per game and 128 in total. Dense-total and Dense-active match the MoE models in total and active parameter counts, and Learned and Hash execute one of four experts per token. Both games are learned solely from transcripts of random legal play, so the information available to the models is



(a) Othello

(b) Chess

Figure 1: Feature arrangements across widths, depths, and FFN structures (move 33, final block). Each point represents one model, with 64 models per game (widths $d \in \{8, 16, 32, 64\}$, depths $L \in \{1, 2, 4, 8\}$, four structures, seed 0). The horizontal axis denotes width, color denotes depth, and marker shape denotes structure. At each width, the four structures are slightly offset horizontally using fixed structure-specific offsets to avoid overlap. The vertical axis shows r, the sum of feature dimensionalities of represented own-piece and opponent-piece directions, measuring the representational capacity occupied by these features and bounded above by d (linear scale from 0 to 1; logarithmic above 1). Separation by color at a fixed width indicates that depth changes the arrangement; intermingled colors indicate that it is already established at depth 1. Dense-total (FFN width 256) and Dense-active (FFN width 64) match the MoE models in total and active parameter counts, respectively. Learned and Hash activate one of four experts, each with width 64.

limited to move sequences generated under the rules. Measurements use moves 17, 25, and 33; the main text reports move 33 at the final block. Training conditions, game counts, and the earlier model sets used in some appendix analyses are described in Appendix A.

4 Results

4.1 Geometry of Internal Representations

Width separates represented features. We compared the number of represented squares, summed dimensional allocation r, and directional overlap across widths for all four FFN structures in both games to assess whether wider models represent more features or separate existing ones. As width increased, r rose and directional overlap decreased, while the number of represented squares changed comparatively little (Figure 1; cosine matrices in Appendix Figure 12). Within the observed range, this suggests that increasing width primarily separates already represented features rather than adding information about new squares. This trend held across all four structures, but differences between structures at a fixed width varied by game. Differences in r were small in Othello, whereas Dense-active and Learned at width 64 had lower r than Dense-total in Chess. Thus, width supports feature separation, but how that width is used in the arrangement also depends on the task and FFN structure.

Depth and FFN capacity support order-dependent features. Differences in feature arrangements at a fixed width motivate a closer look at the computation required to recover each state property. We therefore compared how these representations vary with depth and FFN capacity. Othello occupancy is determined by the initial board and whether each square has received a move, whereas

piece ownership in Othello and own-piece positions in Chess require updating states according to move order. The observed effects of depth and FFN capacity differed accordingly. Othello occupancy was widely represented even at depth 1, with little variation in r across depths and structures at a fixed width (Figure 1a). In contrast, own-piece representations in Chess became prominent at depth 4, and Dense-active, with its smaller FFN capacity, had lower r (Figure 1b). However, r remained small at width 8 even at depth 8, while ownership in Othello became represented at width 64 and depths of at least 4. These results suggest that depth and FFN capacity play a greater role in order-dependent state representations, whose formation also requires sufficient width.

4.2 Model Space and Performance

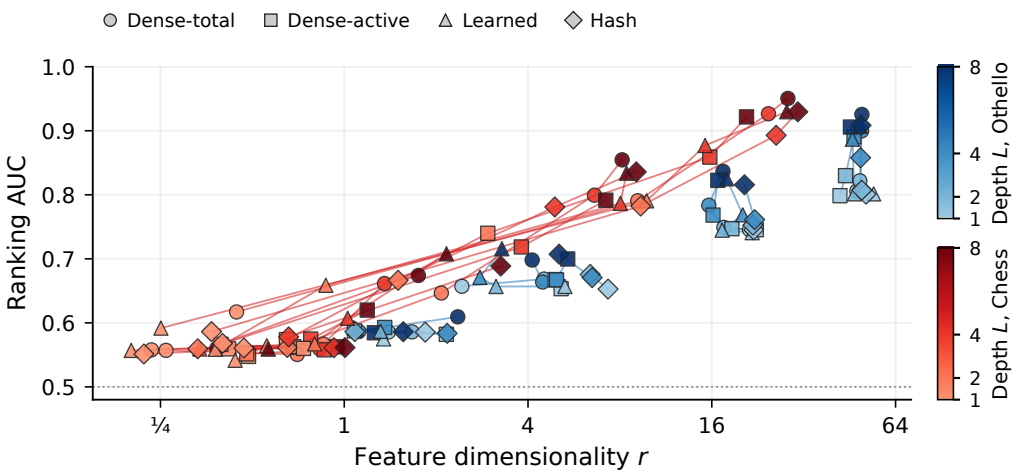
Feature arrangement and output rankings. We compared candidate ranking AUC and legal-move rate across values of r and separated errors into state and rule errors to examine how feature arrangements relate to performance (Figure 2). In Chess, ranking AUC and legal-move rate increased with r, whereas in Othello, performance continued to vary with depth after the arrangement was well established. Error composition reflected this difference. State errors generally decreased with increasing r in both games, while Othello rule errors decreased with depth even where state errors had nearly vanished (Appendix Figures 6 and 7). Within the observed range, these results suggest that performance changes in Chess are closely related to the formation of state representations, while sufficiently wide Othello models additionally benefit from computation that distinguishes legal candidates from the represented states.

A closer look at candidate rankings clarifies this difference. In Othello, deeper models ranked legal candidates more highly even at similar r. Across structures, the ordering by r also differed from the ordering by legal-move rate, which corresponded more closely to ranking AUC. Blockwise output readouts showed that these rankings became pronounced in the final one or two blocks of most models (Appendix Figure 8). Explaining performance therefore requires considering both the size of the arrangement summarized by r and the computation that transforms internal representations into output rankings.

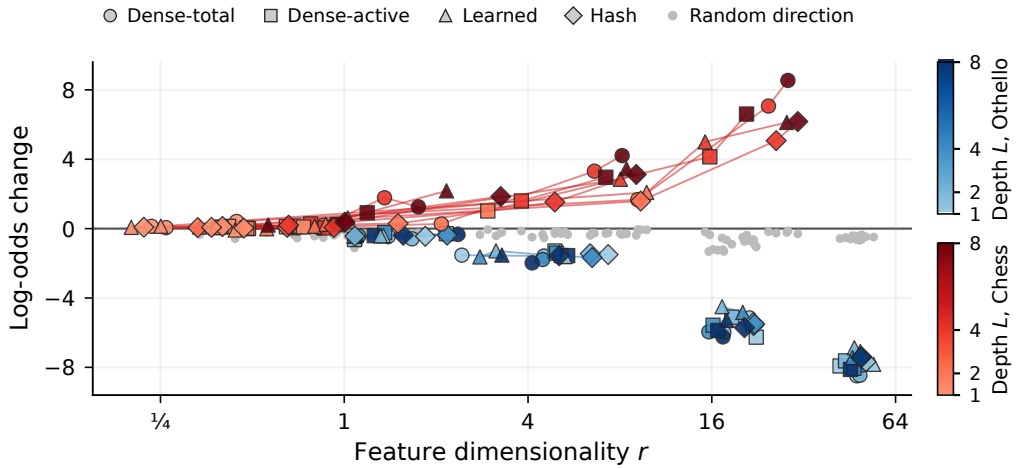
Causal effects of feature directions. These relationships with performance alone do not establish whether the estimated feature directions affect the model's choices. To test this, we added a direction for a specific square to the residual after the final block and measured the change in the log-odds of choosing that square. In Othello, adding an occupancy direction for an empty square lowered its selection log-odds, while in Chess, adding an own-piece direction raised the log-odds of selecting the square as the source (Figure 2b). The magnitude of the change increased with r. In Othello, the effect grew from approximately -0.5 at $r = 1$ to -8 at $r = 50$. In Chess, it grew from nearly zero at $r < 1$ to approximately $+5$ to $+8$ at $r = 20$–30. At similar r, no clear differences emerged across depths or structures. Averaged across models, norm-matched random directions produced effects of -0.5 in Othello and -0.2 in Chess, compared with -3.8 and $+1.4$ for the estimated directions, respectively. These results provide evidence that the estimated directions causally influence output choices in ways consistent with the meanings of the corresponding board properties.

4.3 Towards Empirical Laws of Model Space

The preceding analyses showed that feature arrangements vary with width, depth, and FFN structure, and that performance is related to both these arrangements and the computation that transforms them into outputs. We next examined whether these relationships also appear in next-token cross-entropy. To account for uncertainty in the target distribution across games and stages, we used excess loss $\mathcal{E}$, defined as cross-entropy minus the entropy of the target distribution, and analyzed its relationship with summed dimensional allocation r. Fitting $\mathcal{E} = a - b \log r$ to the 32 Dense-total and Hash models in each game yielded $(a, b) = (1.64, 0.28)$ in Othello and $(1.38, 0.33)$ in Chess, with $R^2 = 0.94$ in both games (Figure 3). Applying the fitted relationship to the 32 Dense-active and Learned models excluded from fitting gave R^2 values of 0.94 and 0.91 and RMSEs of 0.09 and 0.15 nats, respectively. Similar relationships held at moves 17 and 25, with fitting R^2 values of 0.88–0.93 and held-out values of 0.89–0.92 (Appendix E). Within the observed range, the relationship between r and prediction loss therefore extends to structures not used for fitting.



(a) Ranking of legal candidates computed on the arrangement



(b) Causal effect of a square's direction on choosing that square

Figure 2: Candidate rankings and causal effects of feature directions (move 33, 128 models). Blue denotes Othello and red denotes Chess. Lighter shades indicate shallower models, and marker shape denotes structure. (a) Final-block ranking AUC for legal versus illegal candidates, averaged across boards. Candidates are empty squares in Othello and own-piece squares in Chess. The dashed line marks chance (0.5). Blockwise rankings appear in Appendix Figure 8. (b) Change in the log-odds of choosing square s after adding its direction to the final-position residual after the last block and applying the final normalization and output layer. Effects are averaged over cases where the corresponding state is absent, then across squares. Occupancy directions decrease selection in Othello, while own-piece directions increase source-square selection in Chess. Gray points average five norm-matched random directions.

To assess whether this relationship merely reflects nominal model size, we compared the explanatory power of parameter count and width under the same conditions. Using the logarithm of total parameter count gave $R^2 = 0.59$ in Othello and 0.78 in Chess, while the logarithm of width gave 0.97 and 0.43, respectively. Width alone explained loss about as well as r in Othello, but not in Chess. This is consistent with Section 4.1, where Othello arrangements varied mainly with width, while Chess arrangements also depended on depth and structure. Moreover, $\log L$ explained 30% of the variance remaining after the $\log r$ fit in Othello and 8% in Chess. Residuals separated clearly by depth at Othello widths 32 and 64, the same conditions in which output rankings and performance differed at similar r in Section 4.2 (Appendix E). Thus, r summarizes differences in feature arrangements

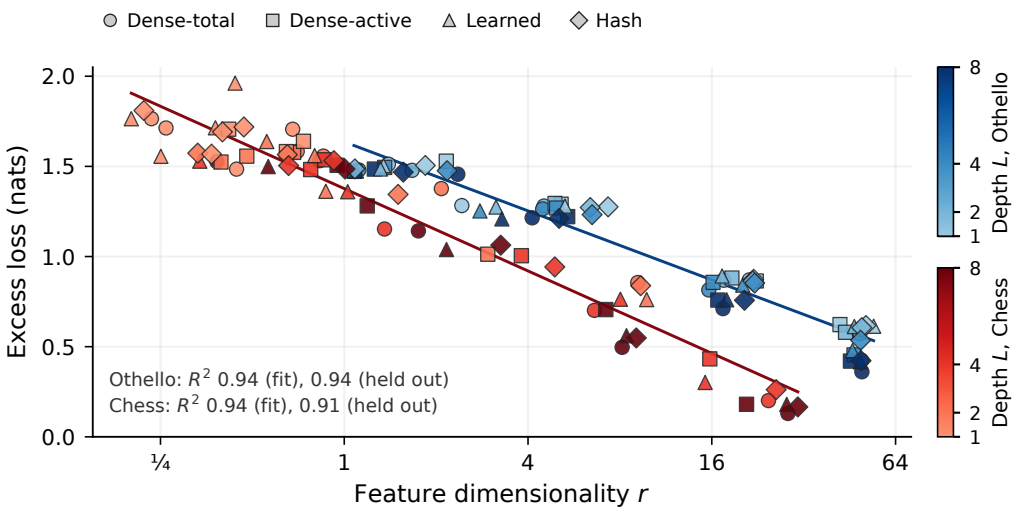


Figure 3: The relationship between excess loss and feature arrangement (move 33, 128 models). The vertical axis shows excess loss: next-token cross-entropy minus the entropy of the target distribution. Othello predicts move tokens and Chess predicts source-square tokens, so their vertical positions are not compared across games. Color denotes the game (blue for Othello and red for Chess), shade denotes depth (lighter: shallower), and marker shape denotes structure. Lines show $\mathcal{E} = a - b \log r$, fitted separately for each game using only the 32 Dense-total and Hash models. Dense-active and Learned models (squares and triangles) are excluded from fitting. Residuals, loss versus total parameter count, and results at moves 17 and 25 are provided in Appendix E.

across widths, depths, and structures, but does not fully account for differences in the computation that maps these arrangements to outputs.

We evaluated predictions whose criteria were fixed before computation to test whether these interpretations hold under additional conditions (items in Appendix F, Tables 3 and 4). Nine of 11 predictions concerning Othello at width 64, new seeds, and width 48 held, as did four of six predictions derived from Othello and applied to Chess (Appendix E). The association between depth gains and reduced rule errors repeated across new seeds and wider models. The reduction in state errors during arrangement formation and the emergence of clear rankings in the final block also held in Chess. Failed predictions, however, revealed conditions under which the relationships did not extend directly. The Dense advantage in Othello color directions appeared only at depth 4, while Chess state errors continued to decrease from depth 4 to 8. At width 48, r exceeded the predicted range, and only two of five separate predictions for the destination-square stage held (Appendix C). These results suggest that extending the empirical relationships requires accounting for the conditions under which new feature families emerge and for differences between prediction stages.

5 Related Works

Board-state representations across models. Prior work has recovered board states from the activations of Othello models trained on move sequences and shown that intervening on internal representations changes move predictions (Li et al., 2023). Encoding pieces as belonging to the current player or the opponent reveals linear state representations (Nanda et al., 2023). Studies of Chess have likewise examined whether models trained on move sequences track board states (Toshniwal et al., 2022; Karvonen, 2024). We apply this encoding across models of different widths, depths, and FFN structures to examine how feature arrangements and error composition vary, rather than only whether square states can be decoded. Motivated by the distinction between information recoverable by linear probes and information needed for a task (Ravichander et al., 2021; Elazar et al., 2021), we distinguish the presence of representations from their use. We assess use through interventions (Vig et al., 2020) by measuring output changes after perturbing a direction and executing the remaining

blocks and output layer. Unlike circuit analyses (Olah et al., 2020; Elhage et al., 2021), we do not separately identify paths through attention and FFN components.

Features, superposition, and the unit of analysis. Toy models have demonstrated how features share limited dimensions according to their importance and sparsity (Elhage et al., 2022). Subsequent work has examined capacity allocation across features (Scherlis et al., 2022) and the relationship between superposition and loss scaling (Liu et al., 2025). SAE studies identify interpretable directions in dictionaries learned to reconstruct activations (Bricken et al., 2023; Cunningham et al., 2023) and investigate how their properties vary with dictionary size and sparsity (Gao et al., 2024; Bussmann et al., 2024). In board games, predefined state properties have also been used to evaluate SAE latents (Karvonen et al., 2024). We analyze rule-defined square-state features and their regressed directions rather than SAE latents, whose correspondence to properties was unstable in our narrow models (Appendix B).

Model size, sparse computation, and performance. Whereas scaling studies predict loss from model and training resources (Kaplan et al., 2020; Hoffmann et al., 2022), we use width and total parameter count as baselines to test whether feature arrangements provide additional explanatory power. MoE models separate total and active parameter counts through expert selection (Shazeer et al., 2017; Fedus et al., 2022; Roller et al., 2021), and toy-model studies have compared feature arrangements and reconstruction across structures (Chaudhari et al., 2025). Our results show that, in Transformers trained to predict moves, executing only a subset of experts can expand or restore feature arrangements, but whether this translates into improved performance depends on the task (Appendix D). These observations are limited to Learned and Hash routers that activate one of four experts and do not establish a general result for other MoE structures.

6 Conclusion and Discussion

We analyzed how model resources and structure relate to feature arrangements and prediction performance through the concept of Model Space. Across both games, increasing width was primarily associated with the separation of already represented features, while the roles of depth and FFN capacity varied with the order dependence of the computation required to recover state information. Differences in candidate rankings and performance at similar dimensional allocations further show that understanding performance requires examining which states are distinguished and how those distinctions reach the output, alongside the size of the representation space. Directional interventions provided evidence that the estimated feature directions affect actual choices. The log-linear relationship between excess loss and summed feature dimensionality extended to structures excluded from fitting, and some predictions held under additional conditions. Together, these findings suggest the potential for conditional empirical laws linking model conditions to performance.

These relationships were observed within a limited range of model sizes and training conditions in two board games. Rule-defined properties and linear direction estimates do not encompass all learned representations, and the MoE results are limited to Learned and Hash routers that activate one of four experts. Although the interventions established effects on outputs, they did not establish whether these directions mediate performance differences across model resources and structures. Failed predictions under additional conditions indicate that the scope of the relationships may depend on the emergence of new feature families and the prediction stage. Extending validation across repeated seeds, wider models, and other tasks, together with identifying paths that mediate performance differences, is the next step toward a more general account of the relationships observed in Model Space.

References

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws, 2024. URL https://arxiv.org/abs/2404.05405.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex

Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bart Bussmann, Patrick Leask, and Neel Nanda. BatchTopK sparse autoencoders, 2024. URL https://arxiv.org/abs/2412.06410.

Marmik Chaudhari, Jeremi Nuer, and Rome Thorstenson. Sparsity and superposition in mixture of experts, 2025. URL https://arxiv.org/abs/2510.23671.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals, 2021. URL https://arxiv.org/abs/2006.00995.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022. URL https://arxiv.org/abs/2209.10652.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022. URL https://arxiv.org/abs/2101.03961.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders, 2024. URL https://arxiv.org/abs/2406.04093.

Wes Gurnee, Emmanuel Ameisen, Isaac Kauvar, Julius Tarng, Adam Pearce, Chris Olah, and Joshua Batson. When models manipulate manifolds: The geometry of a counting task, 2026. URL https://arxiv.org/abs/2601.04480.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Adam Karvonen. Emergent world models and latent variable estimation in chess-playing language models, 2024. URL https://arxiv.org/abs/2403.15498.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models, 2024. URL https://arxiv.org/abs/2408.00113.

Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task, 2023. URL https://arxiv.org/abs/2210.13382.

Yizhou Liu, Ziming Liu, and Jeff Gore. Superposition yields robust neural scaling, 2025. URL https://arxiv.org/abs/2505.10465.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models, 2023. URL https://arxiv.org/abs/2309.00941.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models, 2024. URL https://arxiv.org/abs/2311.03658.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. Probing the probing paradigm: Does probing accuracy entail task relevance?, 2021. URL https://arxiv.org/abs/2005.00719.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. Hash layers for large sparse models, 2021. URL https://arxiv.org/abs/2106.04426.

Adam Scherlis, Kshitij Sachan, Adam S. Jermyn, Joe Benton, and Buck Shlegeris. Polysemanticity and capacity in neural networks, 2022. URL https://arxiv.org/abs/2210.01892.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017. URL https://arxiv.org/abs/1701.06538.

Shubham Toshniwal, Sam Wiseman, Karen Livescu, and Kevin Gimpel. Chess as a testbed for language model state tracking, 2022. URL https://arxiv.org/abs/2102.13249.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Simas Sakenis, Jason Huang, Yaron Singer, and Stuart Shieber. Causal mediation analysis for interpreting neural NLP: The case of gender bias, 2020. URL https://arxiv.org/abs/2004.12265.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints, 2020. URL https://arxiv.org/abs/2002.10689.

A MEASUREMENT DETAILS

Training conditions and model sets. All figures and fits in the main text are based on 128 models spanning two games, four FFN structures (Dense-total, Dense-active, Learned, and Hash), widths $d \in \{8, 16, 32, 64\}$, and depths $L \in \{1, 2, 4, 8\}$, all with seed 0. Each model has four attention heads. The FFN width is 256 for Dense-total and 64 for Dense-active, while Learned and Hash have four experts of width 64 with top-1 routing. Models are trained for 50,000 steps with a cosine schedule and an initial learning rate of 3×10^{-3}; we do not take this as a guarantee of convergence. Some analyses in this appendix come from earlier model sets, which are specified in the figure and table captions. The Othello grid consists of 48 models with $d \in \{8, 16, 32\}$, $L = 1, \ldots, 8$, Dense-total and Hash, and seed 0. Among the Othello Dense-total and Hash cells of the 128-model design, those at widths 8 to 32 are the same models as in this grid. The SAE audit (Appendix B), the initial fit of the loss law and its correction (Appendix F), and pre-registered items A, B, R, and C (Appendix E) come from this grid and its extensions. The extensions are 24 models at width 64 with $L \in \{1, 2, 4, 8\}$ and seeds 0–2, eight models at width 32 with $L \in \{1, 8\}$, both structures, and seeds 1–2, and one four-layer Dense model at width 48 with seed 0. The six existing width-64 models analyzed separately differ from the above: they have four layers and no FFN bias, with three seeds each of Dense and Random.

Chess models are trained on 800,000 games of random legal play, with 30,000 validation games. Each game is 60 plies long. Each token is a square index plus one, so the vocabulary size is 65 and token 0 is unused. A move consists of two tokens, the source square and the destination square, and promotion is restricted to a queen. Activations after k moves are extracted at zero-indexed position $2k - 1$, the position that predicts the next source square. Directions are estimated on validation games $[5000, 10000)$ and evaluated on $[0, 5000)$. The 64 Chess models in the main text are half of the 128-model design above. The destination-square decomposition and pre-registered items P1–P6 (Appendix C) come from an earlier set of 42 models: 18 four-layer models, 18 models at depths 1, 2, and 8 with the same three widths, and six four-layer models at width 64. The two games are compared at moves 17, 25, and 33.

Encoding and detection scope. The common encoding in the main text consists of 128 per-square indicators of own-piece and opponent-piece occupancy (sensitivity to the encoding is examined in Appendix E). The earlier Othello analyses, namely the SAE audit and pre-registered items A–C, used occupancy and color contrasts, which are a rotation of the same 128 dimensions. The auxiliary Chess encoding adds per-square indicators for knight, bishop, rook, queen, and king, giving 448 variables with pawn as the reference piece type. It is not a full one-hot encoding of square states that includes every interaction between color and piece type. With p variables, the denominator 128 in the ridge penalty below is replaced by p. Centering uses the per-current-token means of the fit set, and the same means are applied to the test set. The detection criteria are a predefined stability convention, not significance tests corrected for multiple comparisons. Because ridge selects one representation of the coefficients under the correlations and sum constraints among state contrasts, the geometry can change when the encoding or regularization changes.

Upper bound on the summed feature dimensionality. For a nonzero direction, let $u_i = w_i/\|w_i\|$ and $A = WW^\top$. Then $D_i = \|w_i\|^2/(u_i^\top A u_i)$. Applying the Cauchy–Schwarz inequality on the range of A gives $(u_i^\top A u_i)(u_i^\top A^+ u_i) \geq 1$, so $D_i \leq w_i^\top A^+ w_i$. Therefore,

$$\sum_i D_i \leq \mathrm{tr}(A^+ WW^\top) = \mathrm{rank}(W) \leq d. \tag{6}$$

Because each term is nonnegative, the sum over any particular family of detected directions satisfies the same bound. This bound does not imply that r equals the rank or that the detected directions are independent.

Conditional subspaces. From the covariance of centered states in routing group g, we obtain the projection matrix P_g onto the top $k = d/4$ dimensions. Overlap is the mean of $\mathrm{tr}(P_g P_h)/k$ over the six pairs of groups, and Δ_O is the mean overlap over 20 permutations of the group labels minus the actual overlap. The separate Othello subspace analysis uses 3,000 games each for fitting and evaluation. For Hash, groups are defined by the current token index modulo 4, and the same

partition is applied to Dense. For activation-dependent routers, we compare the actual assignment with a state-dependent partition given by a random projection.

Direction estimation. Activations are the residual at the output of each block, taken before the final RMSNorm is applied. Before centering, the feature vector $f \in \{-1, 0, 1\}^{128}$ consists of the occupancy and color contrasts of the 64 squares. Let X and F denote the activation and feature matrices after subtracting the per-current-token mean from each. We solve

$$\widehat{W} = \arg\min_{W} \|X - FW^\top\|_F^2 + \lambda\|W\|_F^2, \qquad \lambda = 10^{-3}\frac{\mathrm{tr}(F^\top F)}{128}. \tag{7}$$

The random baseline for detection is the 95th percentile of direction norms over 20 regressions in which labels are shuffled across games within the same token. We additionally require the cosine between the directions estimated on two splits to be at least 0.9.

Geometry and decoding. Under the common encoding, the denominator of Equation 4 sums over all 128 directions, whereas the reported occupancy r includes only the contributions of detected occupancy directions. The overlap ratio is computed over pairs of detected occupancy directions for different squares. The AUC of each direction measures how well the projection of token-centered activations from evaluation games onto the unit direction $\hat{W}_i$ distinguishes the corresponding state contrast. Color AUC is computed only over occupied squares. These directions are regression coefficients for reconstructing activations and are not equivalent to linear probes optimized directly for classification performance.

Task metrics. Excess loss is the mean next-token cross-entropy on evaluation games minus the mean entropy of the target distribution. The subtracted entropy $\mathbb{E}[\log n_{\text{legal}}]$ is 2.33, 2.43, and 1.88 nats at moves 17, 33, and 49, respectively. These are not values of the excess loss itself. The empty-square ranking AUC compares the logits of legal and illegal moves among empty squares in the same board position, so it evaluates along a different axis from an AUC that reads the state of a single square across games.

Interventions. At the final token position of a 33-move prefix, we add $\pm W_s$ to the output of the injection block and execute the remaining blocks and the output layer. The effect is measured as the change in the log-odds of the token corresponding to square s, with five random directions of equal norm as controls. The direction for each injection site is estimated separately at that block. Differences in effects across sites therefore combine differences in direction, magnitude, activation distribution, and subsequent computation. Injecting the direction estimated at a middle or the first block at its natural magnitude has almost no effect in either game, because direction norms at those sites are below one hundredth of those at the final block; we do not read this as a decomposition into direct and indirect effects.

B SAE AUDIT

We trained a total of 240 BatchTopK dictionaries with $K \in \{d/8, d/4, d/2, d, 2d\}$ on the final block of the 48 grid models and evaluated them with 308 rule-based properties, drawing on the board-property approach to semantic evaluation of Karvonen et al. (2024). However, the precision threshold and match counts below are not the same as that work's definition of coverage or its board reconstruction metric. The number of active latents per dictionary is 12–14, about 25, and 51–59 at widths 8, 16, and 32, respectively. After merging pairs of opposite sign, the number of axes is 9.5–12.5, about 17, and about 33, respectively (Figure 4). The number of latents that match a property with precision of at least 0.95 is largest at $K = d/4$ and nearly zero at $K = d$, and the matched properties are limited to occupancy and frontier properties.

The axis clusters and F1 comparisons in the figure are audit data from the $K = d/2$ and $K = d$ conditions. For all five property families, the F1 of an individual SAE latent differs little from the current-token baseline, by roughly 0.05 or less. In contrast, linear probes on all latents improve with width, as do probes on raw activations. Semantic correspondence and usefulness for reconstruction also differ. In a separate K grid, loss recovered at $K = d/8$ is negative, so reconstructions in this setting do not preserve the model's predictions. The number of latents linked to properties with high precision is therefore not sufficient on its own to assess the amount of task-relevant representation.

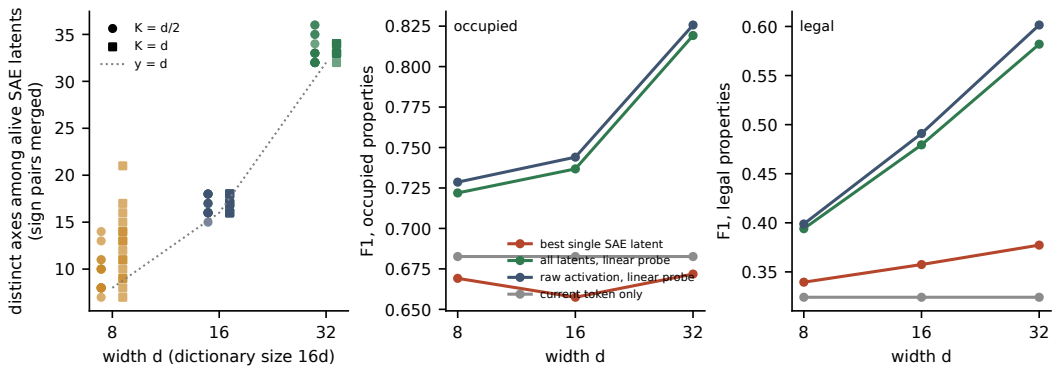


Figure 4: SAE evaluation. Left: the number of axes formed by grouping active latents of BatchTopK SAEs (dictionary size $16d$) trained on the final block at $|\cos| \geq 0.9$, with the reference line $y = d$. Middle and right: F1 for occupancy and legality properties. Red denotes the best individual SAE latent, green a linear probe on all latents, blue a linear probe on raw activations, and gray the current-token baseline.

C Chess comparisons and measurement boundaries

Definitions for the behavioral decomposition. The top choice is the argmax over square tokens 1–64 in the stored logits. We reconstruct the board with python-chess and classify the chosen square as a legal source square, a square with an own piece but no legal move, a square with an opponent piece, or an empty square. The ranking AUC among own pieces is the AUC with which logits distinguish legal source squares from the other own-piece squares within a board, averaged over boards; only boards that contain both classes are used. Excess loss is the cross-entropy of the source-square token minus $H(\text{from}) = -\sum_s p_s \log p_s$, where $p_s =$ (number of legal moves starting from square s)/(total number of legal moves). At moves 17, 25, and 33, $H(\text{from})$ is 2.32, 2.27, and 2.20 nats. The logit lens computes the same ranking AUC from logits obtained by applying the final RMSNorm and unembedding to the residual after each block. The six predictions and their evaluation criteria were fixed in a file, together with its hash, before computation.

Occupancy and color. In the four-layer Dense models of the primary Chess set at width 32, the number of detected occupancy directions averages 36, 42, and 45 at moves 17, 25, and 33, and occupancy r is 2.2, 2.6, and 3.0. A regression on 128 owner-specific variables from the same activations gives r_{state} of 5.4, 6.0, and 6.7. At move 33, own pieces contribute 6.664 and opponent pieces 0.006. Opponent directions are detected for four squares on average, so we do not equate a small contribution with zero detections. Results for the 448 variables that add piece types also use a different denominator and are not combined directly. In addition, because the source-square distribution is weighted by the number of moves available from each square, we do not subtract the entropy over all legal moves from the source-token cross-entropy.

To-square decomposition. At the position where the actual source-square token is given, we classify the model's top destination-square choice as legal, a square the piece cannot reach even on an empty board, a state error in which the destination square holds an own piece or a pawn's diagonal has no piece to capture, a blocked path, or a move that leaves the king under attack. For the width-32 Dense model at move 33, at depths 1, 2, 4, and 8, legal destination squares account for 68, 86, 93, and 96%, unreachable-square errors for 20.8, 5.2, 1.8, and 0.3%, and state errors for 8.9, 7.2, 3.1, and 1.8%; blocked paths stay below 0.4%, and king-safety errors range from 1.7 to 1.9% (Figure 5). The ranking AUC among squares the piece can reach is 0.81, 0.86, 0.91, and 0.94. Errors that choose a square the piece cannot reach even on an empty board fall from 21% to 0.3% between depths 1 and 8 and account for most of the gain. State errors with an own piece on the destination square come next, while moves that leave the king under attack stay flat at around 2% across all widths and depths. While the source-square stage requires the arrangement of own pieces, the destination-square stage requires piece types and movement rules, and models at this scale do not learn to avoid check. The excess loss of the destination token is its cross-entropy minus $\log$(number of legal desti-

nation squares for the piece), and the excess loss of a two-token move is the sum of the two values. We fixed five items before computation (Table 1), and two of them held. Q1 and Q2, which predicted that depth gains would come from reductions in state errors, failed because most of the reduction came from unreachable-square errors. Q5, which predicted that the two-token loss would fit $\log r$ with R^2 of at least 0.9, failed because R^2 was 0.88 at move 17 and $\log d$ gave a higher value.

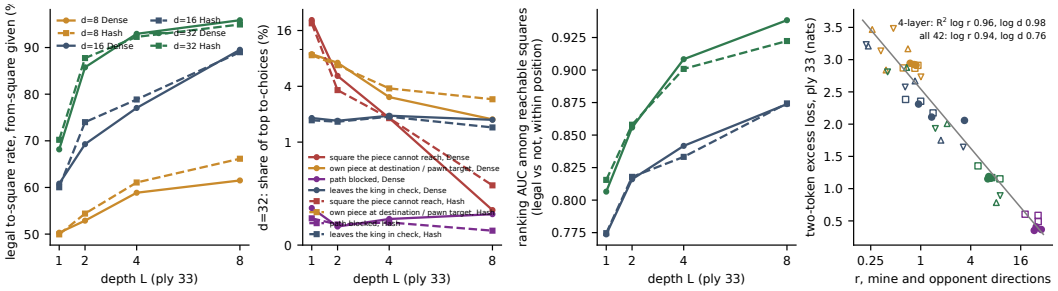


Figure 5: The Chess destination-square stage (after the actual source square is given, move 33). From left: the rate of legal destination squares, error composition at width 32, ranking AUC among reachable squares, and the relationship between the excess loss of two-token moves and r (circles and squares: four layers; triangles: depths 1, 2, and 8; gray line: fit to the four-layer models).

Table 1: Five items fixed before computation for the Chess destination-square stage.

Item	Fixed prediction	Result
Q1	Depths 1 to 4: share of state-error reduction at least 0.5	Failed: 0.24, 0.22
Q2	Depths 4 to 8: share of state-error reduction at least 0.5	Failed: 0.43, 0.37
Q3	King-safety errors at most 3% at all depths	Held: 1.4–2.3%
Q4	Ranking AUC among reachable squares: increase of at least 0.05 from one to four layers and at least 0.02 from four to eight layers	Held: +0.10/+0.09, +0.03/+0.02
Q5	Four layers, widths 8–64: $b > 0$ for the two-token loss and $R^2 \geq 0.9$ for $\log r$	Failed: $b > 0$, but R^2 is 0.88 at move 17 and higher for $\log d$

Conditional subspaces across blocks. Averaged over three seeds of the four-layer primary Chess models, Δ_O at the third block is 0.205, 0.088, and 0.114 for Dense and 0.331, 0.229, and 0.222 for Hash at widths 8, 16, and 32. At the final block, the corresponding values are 0.002, 0.001, and 0.011 for Dense and 0.010, 0.042, and 0.026 for Hash.

D Partial expert execution with Hash routing

Partial expert execution and the separation of feature directions. In Othello, Hash has larger r and a smaller overlap ratio than Dense at widths 16 and 32 (6.5 vs. 4.5 and 22.1 vs. 15.6 at four layers and move 33). In Chess, Hash has half the r of Dense at width 16 (1.0 vs. 1.9) and similar r at widths 32 and 64. Because the separation of the arrangement varies with game and width even at the same sparsity s, the number of executed experts does not determine the separation of the representation space.

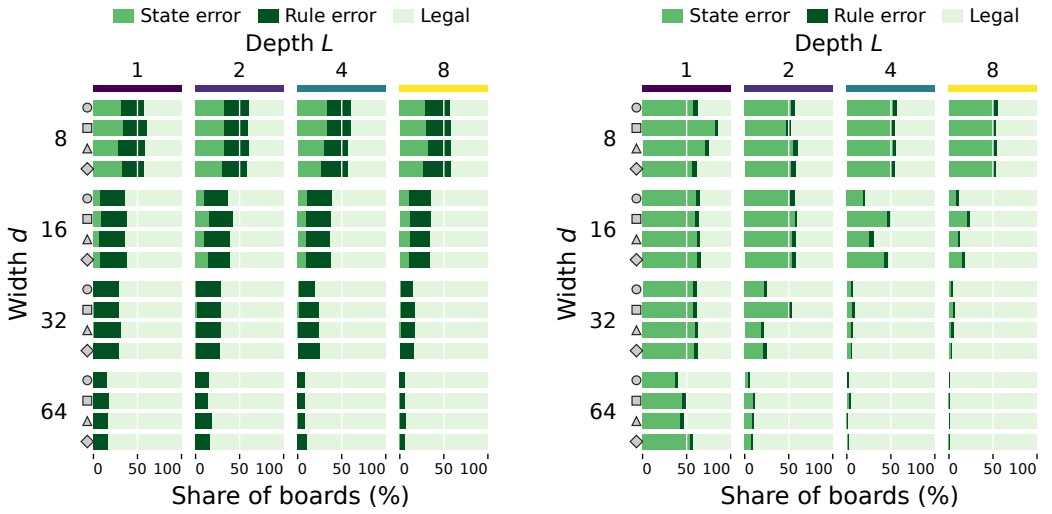
Separation and performance under Hash routing. At width 32 with four layers in Othello, Hash has larger r than Dense, but its legal-move rate at move 33 is 74%, lower than the 80% of Dense. At width 16 with four layers in Chess, Hash has half the r of Dense, and its legal-move rate for completed moves is 12 to 16 percentage points lower in all three seeds; the difference lies in state errors that choose an empty square or an opponent piece as the source square. The same structural difference increases separation without improving performance in one game, and lowers both separation and performance in the other. The amount of separation alone cannot predict performance; the error types must also be examined to identify which component changed.

E ROBUSTNESS AND PRE-REGISTERED CHECKS

Sensitivity to the feature encoding. The 128 features in the main text are own-piece and opponent-piece indicators. The occupancy ($\text{mine}_s + \text{opponent}_s$) and color ($\text{mine}_s - \text{opponent}_s$) contrasts commonly used in Othello analyses are a rotation of the same 128 dimensions, and at move 33 across the 48 Othello grid models, the total r under the two encodings agrees with a correlation of 0.9999 (means of 8.7 vs. 8.6). In Chess, fixing the move fixes the player to move, so occupancy and color become nearly the same variable on home-rank squares; the joint regression splits a single direction between the two coefficients, which then fail the stability criterion. At width 32 with four layers, r is 3.0 under the contrast encoding and 6.7 under the indicator encoding. Both encodings represent the same label space but do not preserve per-direction detection or the sum of D_i, so the main text uses the indicator encoding throughout.

Othello seed replication and extrapolation to width 64. In eight newly trained models with one and eight layers at width 32 and seeds 1 and 2, the legal-move-rate gain of eight layers over one layer is 13.4 to 14.5 percentage points in all four pairs, and the share of the gain at move 33 due to reduced errors on illegal empty squares is 1.03 to 1.16. In the 24 width-64 Dense and Hash models trained under the same conditions, errors that choose an occupied square are at most 0.03% at every depth, and the depth gain has the same error composition. Substituting $r \approx 45$ at width 64 into Equation 5 fitted to the 48 grid models predicts excess losses of 0.78, 0.62, and 0.32 at moves 17, 33, and 49, whereas the observed values are 0.45, 0.39, and 0.33. The prediction is close late in the game and overestimates the loss at earlier moves, which coincide with the moves at which color directions first become represented at width 64.

Composition of top choices. Figure 6 is the basis on which Section 4.2 distinguishes regimes by error type; it divides the top choices of the 128 models into state errors, rule errors, and legal choices.

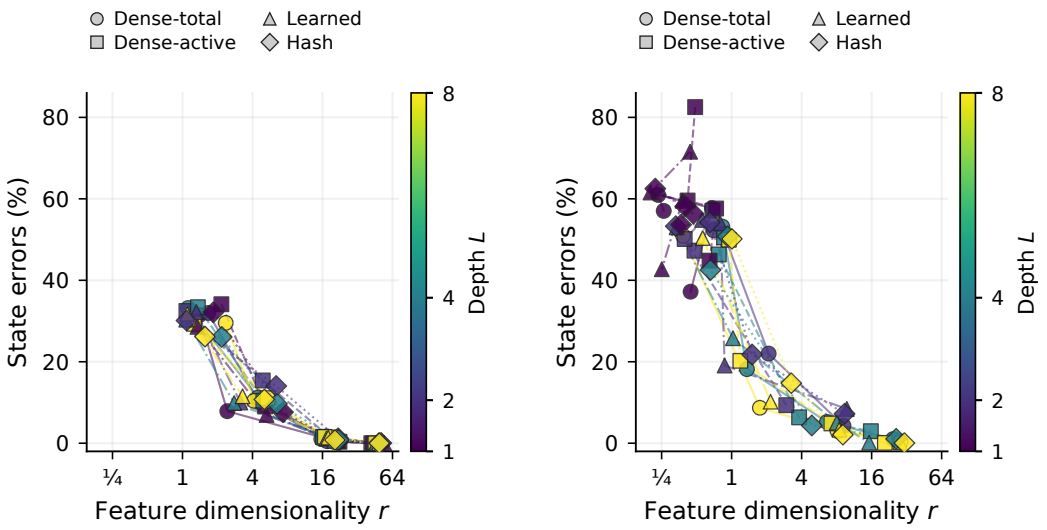


(a) Othello: composition of top choices

(b) Chess: composition of top from-square choices

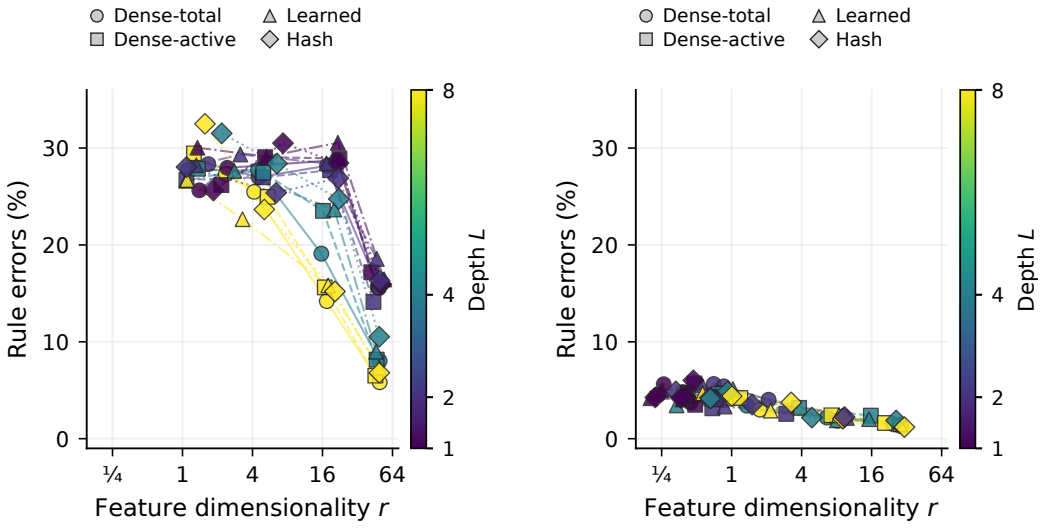
Figure 6: Composition of top choices (move 33, 128 models). Each bar covers all evaluation boards of one model (100%) and divides the top choices into state errors, rule errors, and legal choices. State errors are choices inconsistent with the board state (an already occupied square in Othello; an empty square or an opponent-piece square chosen as the source square in Chess), and rule errors are choices consistent with the state but not allowed by the rules. For Chess, a legal choice means a legal source square. Rows denote width and structure, and columns denote depth; marker shapes and depth colors follow Figure 1. The white line in the middle of each bar marks 50%. Large state errors indicate that the arrangement limits performance, whereas rule errors that decrease with depth in rows where state errors have vanished indicate that computation over the arrangement limits performance. The same errors are plotted against r in Figure 7.

Error rates against the summed feature dimensionality. Figure 7 replots the two errors of Figure 6 against the r of each model and supports the statement in Section 4.2 that state errors decrease with r.



(a) Othello: state errors against r

(b) Chess: state errors against r



(c) Othello: rule errors against r

(d) Chess: rule errors against r

Figure 7: The two errors of Figure 6 against r (move 33, 128 models; markers and colors as in Figure 1). The horizontal axis is the r of each model. Thin lines connecting the models of widths 8, 16, 32, and 64 with the same structure and depth show the order of experimental conditions, not fits or predictions. State errors that decrease with r regardless of color indicate that the arrangement limits performance, whereas rule errors that separate by color where state errors are near zero indicate that computation over the arrangement limits performance.

An exception to the state-error curve. In Chess at width 16, depth 8 reduces state errors from 61% to 13% while r remains between 0.3 and 2.1 (averaged over the four structures). The arrangement measured by linear directions alone cannot explain this change in performance.

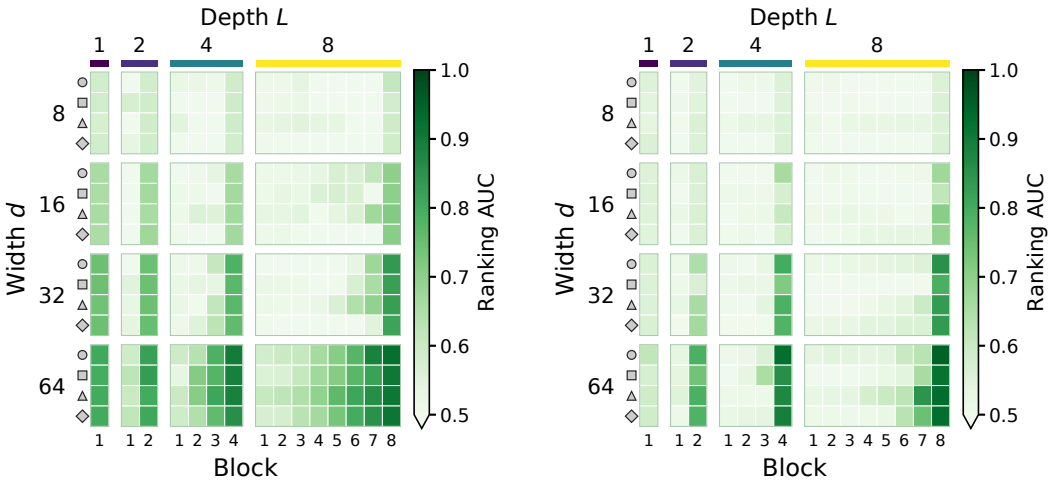
Colour information readable by linear and non-linear probes. Given that rule errors may include failures to represent color (Section 4.2), we checked whether color information that is not

captured linearly becomes more readable with depth. In Table 2, an MLP with two hidden layers and per-square MLP classifiers read color no better than the linear probe (differences from -0.03 to $+0.01$). Meanwhile, the color information readable by all three probes increases with depth. At width 32, it rises from 0.50 at depth 1 to 0.59 at depth 8; at width 64, it rises from 0.51 to 0.66 at depth 4 and 0.72 at depth 8, matching the range in which rule errors decrease with depth. This check supports only the conclusion that color information readable by these probes increases with depth; whether the model uses that information to reduce rule errors is a separate question.

Table 2: AUC of probes that read the color of occupied squares (own vs. opponent disc) from final-block activations in Othello (move 33, test games, averaged over four structures). Linear is the ridge probe of the main text, MLP is a regression with two hidden layers (each of width $4d$), and MLP per square is a per-square classifier with one hidden layer trained only on positions where the square is occupied. All probes are trained on the same fit games and evaluated on the same test games.

d	Probe	$L = 1$	2	4	8
32	linear	0.50	0.50	0.52	0.59
32	MLP $2 \times 4d$	0.50	0.50	0.51	0.59
32	MLP per square	0.51	0.50	0.51	0.56
64	linear	0.51	0.52	0.66	0.72
64	MLP $2 \times 4d$	0.51	0.51	0.64	0.70
64	MLP per square	0.51	0.51	0.65	0.72

Where the ranking is computed. Figure 8 shows the ranking AUC measured with a logit lens that applies the final normalization and output layer to the intermediate residual after each block. In most models, the ranking becomes pronounced in the final one or two blocks, and 31 of the 32 eight-layer models reach 90% of their final ranking (measured as the margin above 0.5) only at the final block. This means that candidate rankings readable through the final output layer become pronounced late in the computation, not that the computation in earlier blocks is unnecessary.



(a) Othello

(b) Chess

Figure 8: Candidate rankings read at each block (move 33, 128 models). Rows denote width and structure, and columns are the blocks of each model grouped by depth, with the rightmost column of each group being the final block. The green shade of each cell shows the same ranking AUC as in Figure 2, and values of 0.5 or below are shown in the lightest green.

Depth term, model size, and other plies of the loss law. Plotting the residuals of the main-text equation $\mathcal{E} = a - b \log r$ against width (Figure 9a, b) shows that residuals line up in order of depth

only at Othello widths 32 and 64. The mean residual is $+0.09$ and $+0.05$ nats at depth 1 and -0.09 and -0.16 nats at depth 8, and $\log L$ explains the residuals with R^2 of 0.76 and 0.85. At widths 8 and 16, the corresponding R^2 values are 0.07 and 0.13, and in Chess they do not exceed 0.30 at any width. Fitting $\log r$ and $\log L$ jointly to all 128 models gives a $\log L$ coefficient of -0.064 in both games and raises R^2 from 0.943 to 0.960 in Othello and from 0.928 to 0.934 in Chess. In Othello, a two-variable equation in $\log d$ and $\log L$ reaches R^2 of 0.98 because the r of Othello is a function of width. Plotting the same models against total parameter count (Figure 9c, d) shows that Othello models form a band at each width: although the parameter count increases 16- to 23-fold within a width, the range of excess loss is only 0.08 nats at widths 8 and 16 and 0.26 nats at width 64. In Chess, the range within a width is 0.6 to 1.4 nats. At moves 17 and 25 as well, the equation fitted to Dense-total and Hash transfers to Dense-active and Learned (Figure 10). The fitted R^2 values are 0.88 and 0.92 in Othello and 0.92 and 0.93 in Chess, and the held-out values are 0.89 and 0.92 in Othello and 0.92 and 0.90 in Chess.

Pre-registered items. The 11 Othello items (A1–A3, B1–B3, R1–R2, C1–C3) are listed in Table 3, and the six Chess items (P1–P6) in Table 4. Figure 11 shows the contents of the checks. A and B test previously unanalyzed measurements of width-64 models whose overall performance was already known, R tests replication with new seeds, and C tests interpolation to a new intermediate width. For C, the training logs printed the legal-move rate, so the plan to keep performance hidden until the features were measured was not fully followed. The six Chess items were fixed with the legal-move rates and r of the models already known, but before error types, rankings, logit lens results, and excess loss were computed. The items are not independent statistical tests, and the number of items that held should not be interpreted as a probability of successful generalization.

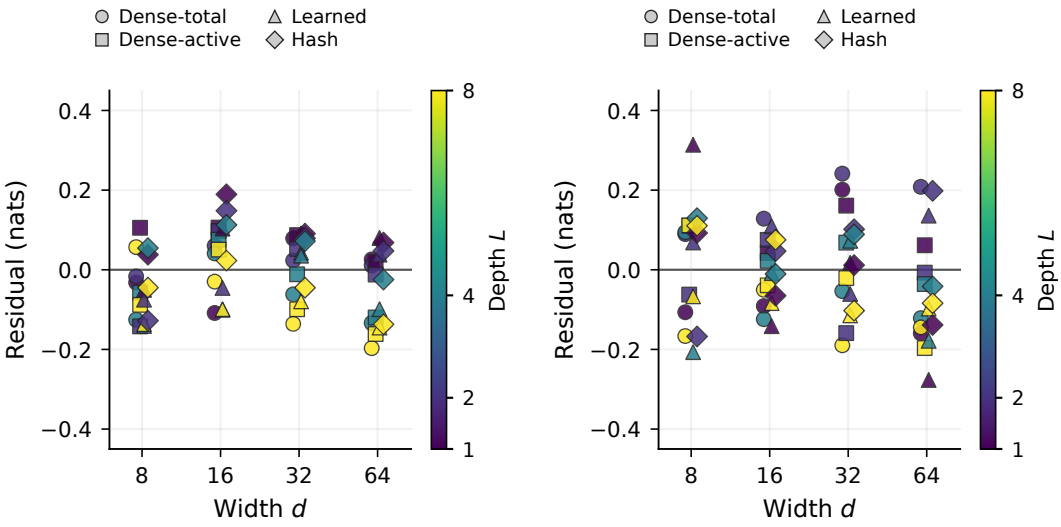
F Additional geometry and prediction provenance

Correction of the loss-fit coefficients. The initial fit contained 60 rows because the rows for one and two layers were duplicated. After the correction, we recomputed the coefficients from the 48 unique models. At $r \approx 45$, the originally recorded predictions were about 0.77, 0.65, and 0.37 nats at moves 17, 33, and 49, and the corrected predictions are about 0.78, 0.62, and 0.32. The goodness of fit and figures in the main text are based on the corrected coefficients, and we do not present the recomputation as a new advance prediction. The measured excess losses of the existing width-64 Dense models are about 0.45, 0.39, and 0.33, and those of Random are 0.58, 0.48, and 0.37.

Verdict units and strength of validation. The registration document contains 11 individual items: A1–A3, B1–B3, R1–R2, and C1–C3. In the verdict file, A1 and C1 failed and the rest held. An earlier summary reported seven of nine, which does not match the individual verdicts; we have corrected it in this manuscript. We do not count the two structures in B or the four comparisons in R as additional independent predictions. Verdicts from a separate conditional-occupancy registration are not mixed into this set.

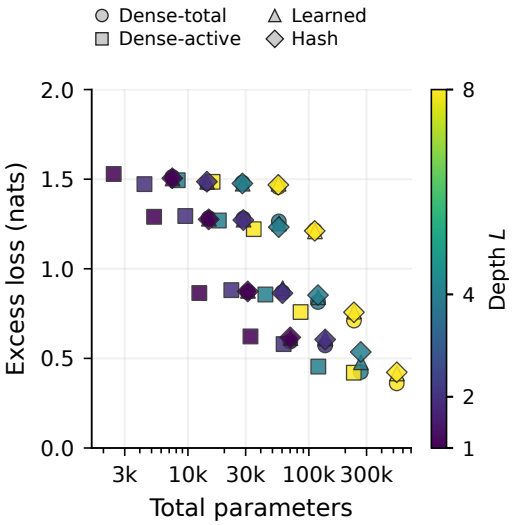
A and B are additional tests on previously unanalyzed measurements of models whose overall performance was known, R is replication with new seeds, and C is interpolation to a new intermediate width. For C, the training logs printed the legal-move rate, so the plan to keep performance hidden until the features were measured was not fully followed. Because the branch of C3 with ten or more detected color directions did not occur, the hypothesis that the width baseline fails when a new feature family first becomes represented has not yet been tested prospectively. At width 48, r is 34.71, 36.69, and 34.33 at moves 17, 33, and 49, and no color directions are detected at move 17. The residuals of the width baseline are -0.016, -0.034, and -0.045 nats.

Comparison scales and the division of roles. In the legal-move rate over 59 positions, the gain of eight layers over one layer for Dense is 14.9 percentage points at width 32 and 10.4 percentage points at width 64, whereas the relative reduction in the remaining errors is about 56% and 72%. Because the comparison scale was not specified before the results were seen, we do not draw the one-directional conclusion that depth gains are larger at wider widths. Likewise, the dependence of color detection on depth at width 64 does not support an exclusive division of roles in which width determines only representation and depth determines only computation.

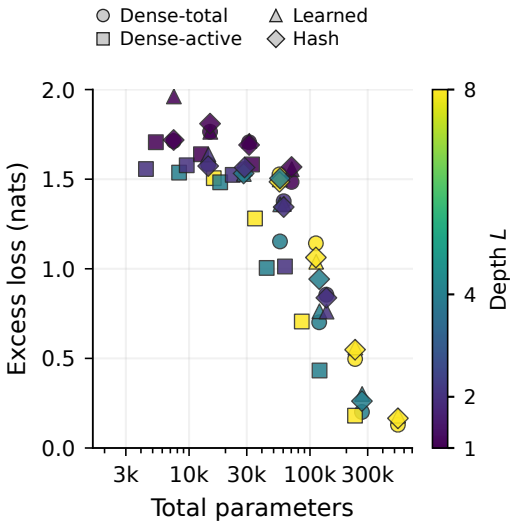


(a) Othello: residual of the law against width

(b) Chess: residual of the law against width

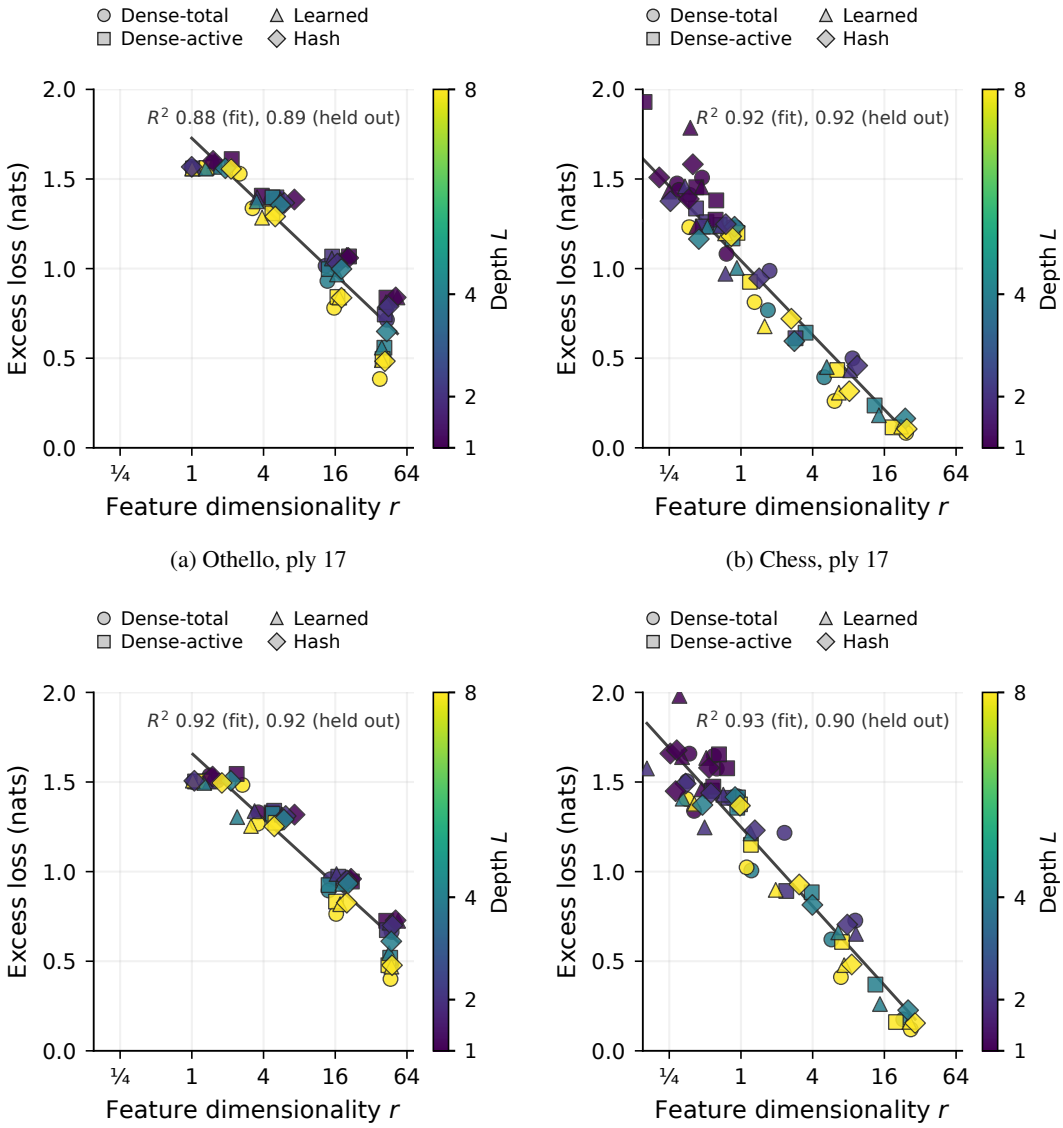


(c) Othello: excess loss against total parameters



(d) Chess: excess loss against total parameters

Figure 9: Residuals of the equation in Figure 3 and model size (move 33, 128 models; markers and colors as in Figure 1). (a, b) Residuals $\mathcal{E} - (a - b \log r)$ plotted against width, with the four structures at each width offset by fixed intervals. Residuals ordered by color at the same width indicate that depth changes the loss beyond r. (c, d) Excess loss of the same models plotted against total parameter count.



(a) Othello, ply 17

(b) Chess, ply 17

(c) Othello, ply 25

(d) Chess, ply 25

Figure 10: Excess loss against r at moves 17 and 25. As in Figure 3, lines show the equation fitted only to Dense-total and Hash, and squares and triangles are excluded from fitting.

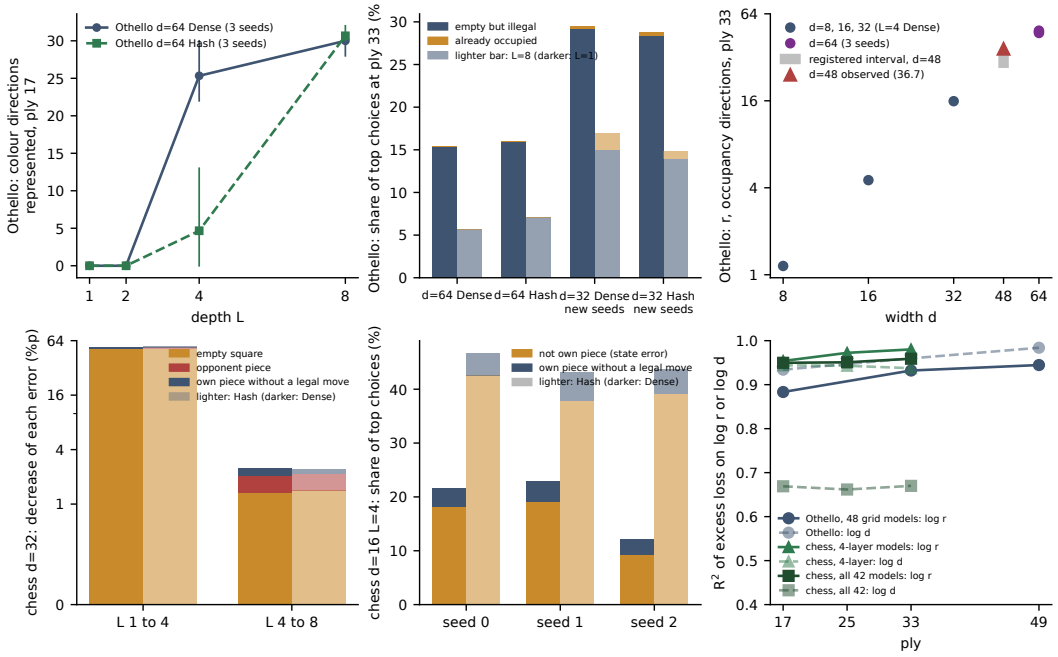


Figure 11: Checks with criteria fixed before computation. Top: Othello. Left: detected color directions at move 17 for width-64 Dense and Hash models across depths, averaged over three seeds with min–max ranges. Dense leads only at depth 4 (A1 failed, A2 held). Middle: top-choice errors at move 33 for depths 1 and 8 (left and right bars). Depth mainly reduces empty-but-illegal choices (B, R). Right: r versus width at move 33. The width-48 observation (triangle) exceeds the registered range (gray band, C1 failed). Bottom: Chess. Left: error reductions at width 32 from depths 1 to 4 and 4 to 8. State errors dominate the first interval (P1 held), while rule errors account for less than half of the reduction in the second (P2 failed). Middle: errors across seeds at width 16 and depth 4. The Hash deficit lies in state errors (P4 held). Right: R^2 for excess-loss fits using $\log r$ and $\log d$ in both games (P6 failed).

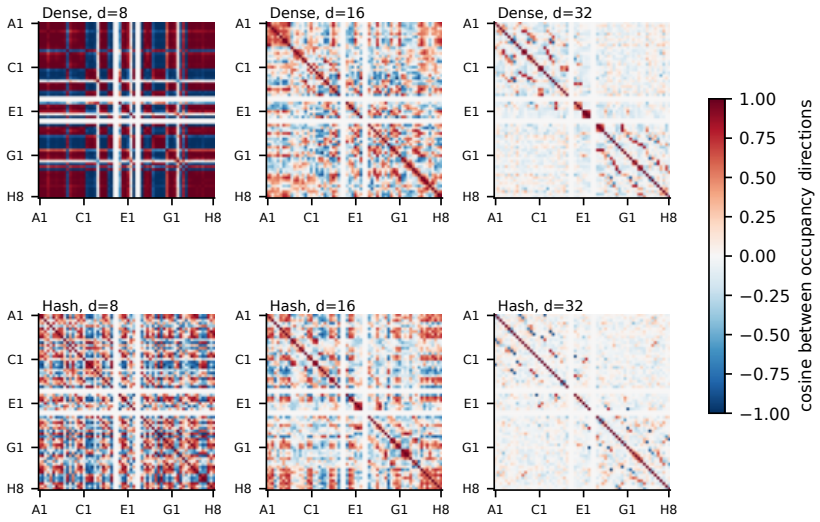


Figure 12: Cosine matrices of occupancy directions at the final block in Othello ($L = 4$). Top: Dense; bottom: Hash; columns: widths 8, 16, and 32. Narrow models share directions, whereas wider models separate them relatively more.

Table 3: The 11 items whose evaluation criteria were fixed in advance. A and B test previously unanalyzed measurements of existing width-64 models, R tests new seeds, and C tests a new width. The items are not independent statistical tests, and the number of items that held should not be interpreted as a probability of successful generalization.

Item	Fixed prediction	Result
A1	Number of color directions detected at move 17: Dense $>$ Hash at three or more of four depths	Failed: held only at four layers
A2	Color-direction strength and AUC: Dense $>$ Hash at three or more depths for each	Held: 4/4 and 3/4, respectively
A3	Legal-move-rate gap between structures: move 17 $>$ move 49 at both four and eight layers	Held: 5.3/2.2, 2.7/0.8 pp
B1	Occupied-square errors: at most 1.5% at all depths	Held: at most 0.033%
B2	Share of the depth gain from reduced empty-square errors: at least 0.8	Held: 1.000, 0.996
B3	Depth gain: move 17 $>$ move 49	Held: 18.5/6.9, 17.8/6.1 pp
R1	Legal-move-rate gain of eight layers over one layer for two new seeds: at least 8 pp	Held: 13.4–14.5 pp in four comparisons
R2	Share from reduced empty-square errors for new seeds: at least 0.8	Held: 1.03–1.16
C1	r at move 33 for width 48: $[27, 35]$	Failed: 36.7
C2	Residual of the width baseline at move 49: within ± 0.05 nats	Held: -0.045
C3	If at least 10 color directions are detected at move 17, residual < -0.05; otherwise within ± 0.05	Only the latter held: 0 detected, -0.016

Table 4: Six items fixed before computation for Chess. Move 33; at width 32, four-layer results are averaged over three seeds and other depths use seed 0.

Item	Fixed prediction	Result
P1	Width 32, depths 1 to 4: share of reduced state errors (empty square, opponent piece) at least 0.5 for both structures	Held: 0.97, 0.96
P2	Width 32, depths 4 to 8: share of reduced rule errors at least 0.5 for both structures	Failed: 0.18, 0.12
P3	Width 32, ranking AUC among own pieces: increase of at least 0.02 from four to eight layers and at least 0.05 from one to four layers	Held: +0.047/+0.034, +0.245/+0.234
P4	Width 16, four layers: Hash state errors $>$ Dense in all three seeds	Held: 42.5/37.7/39.2 vs. 18.2/19.1/9.2%
P5	Width 32, eight layers, logit lens: ranking AUC above 0.6 only at the final block	Held: $\leq$ 0.56 through block 7; 0.86/0.84 at block 8
P6	Four layers, widths 8–64: $b > 0$ and R^2 of $\log d \geq R^2$ of $\log r_{\text{state}}$	Failed: $b > 0$, but $\log r_{\text{state}}$ is higher (0.95/0.97/0.98 vs. 0.94/0.94/0.94)