Superposition in Mixture of Experts

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Superposition allows neural networks to represent far more features than they have dimensions. Previous work has explored how superposition is affected by attributes of the data. Mixture of Experts (MoE) models are used in state-of-the-art large language models and provide a network parameter that affects superposition: network sparsity. We investigate how network sparsity (the ratio of active to total experts) in MoEs affects superposition and feature representation. We extend Elhage et al. [2022]'s toy model framework to MoEs and develop new metrics to understand superposition across experts. Our findings demonstrate that MoEs consistently exhibit greater monosemanticity than their dense counterparts. Unlike dense models that show discrete phase transitions, MoEs exhibit continuous phase transitions as network sparsity increases. We define expert specialization through monosemantic feature representation rather than load balancing, showing that experts naturally organize around coherent feature combinations and maintain specialization when initialized appropriately. Our results suggest that network sparsity in MoEs may enable more interpretable models without sacrificing performance, challenging the view that interpretability and capability are fundamentally at odds.

1 Introduction

Mixture of Experts (MoEs) have become prevalent in state-of-the-art language models, such as Qwen3, Mixtral, and Gemini [Yang et al., 2025, Jiang et al., 2024, Google DeepMind, 2025], primarily for their computational efficiency and performance gains [Shazeer et al., 2017, Fedus et al., 2022]. However, despite their widespread adoption, MoEs remain poorly understood from a mechanistic interpretability perspective.

A fundamental challenge in interpreting neural networks is the phenomenon of superposition: when models represent more features than they have dimensions. This allows networks to pack many sparse features into fewer neurons at the cost of making individual neurons polysemantic and difficult to interpret.

MoE architectures introduce a new dimension to this problem: network sparsity. Unlike dense models that activate all neurons regardless of input, MoEs activate only a small fraction of their total parameters [Shazeer et al., 2017]. While dense models exploit feature sparsity by packing many sparse features into shared neurons, MoEs can afford to be more selective, potentially dedicating entire experts to specific feature combinations.

We investigate whether (1) MoEs exhibit less superposition than their dense counterparts, (2) there is a discrete phase change in MoE experts as seen in dense models, and (3) we can understand expert specialization through the lens of feature representation rather than just load balancing.

We explore these questions using toy models that extend Elhage et al. [2022]'s framework to MoEs. Our key contributions are as follows: (1) MoEs consistently exhibit greater monosemanticity (less superposition) than dense models with equivalent active parameters, with individual experts

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

representing features more cleanly; (2) unlike dense models, MoEs do not exhibit sharp phase changes, instead showing more continuous transitions as network sparsity increases; and (3) we propose an interpretability-focused definition of expert specialization based on monosemantic feature representation, showing that experts naturally organize around coherent feature combinations rather than arbitrary load balancing.

These findings suggest that MoEs may offer a path toward more interpretable architectures—achieving better performance while maintaining more monosemantic feature representations. This work provides both theoretical insights into how network architecture affects superposition and practical tools for analyzing these increasingly prevalent models.

2 Background

We expand upon the toy model setup formalized by Elhage et al. [2022]. Consider an autoencoder with n input features and m hidden dimensions trained to minimize the reconstruction loss $(x - \hat{x})^2$:

$$\hat{x} = \text{ReLU}(W^T W x + b) \quad \text{where } x, \hat{x} \in \mathbb{R}^n, W \in R^{n \times m}, b \in \mathbb{R}^n$$

We construct toy MoEs that replicate this architecture for each of e experts, where each matrix W_i^e is indexed by its expert for the ith feature vector. The input x is routed to the top-k experts calculated by taking softmax of $(W_r x)$, where $W_r \in R^{n \times E}$ is the gate matrix. We take top-$k = 1$ to simplify these toy models and match the active parameters between MoE and dense models.

We sample x such that each feature x_i has feature sparsity ($S \in (0, 1]$, or feature density given by $1 - S$) and the last feature rx_{-1} has relative importance ($r \in \mathbb{R}^+$). Feature sparsity governs the likelihood a particular input feature dimension is zero. The relative importance is a scalar on the magnitude of the last feature, so $x \in \{x_1, x_2, ...rx_n\} : x_i \in U(0, 1)$ with S likelihood that $x_i = 0$.

We define network sparsity as the ratio of total active experts (top-k) to the total number of experts, E. Because top-$k = 1$, the network sparsity is given by $1/E$ and is completely governed by E; a network with one expert is equivalent to a 'dense', non-MoE model. The input dimensions n scales the input vector size, while the hidden dimensions m allows us to control the representational capacity of the networks. We do not use a load balancing loss unless otherwise noted in order to do a fair comparison with the dense models.

To demonstrate superposition in various models, we use two key visualizations as defined by Elhage et al. [2022]. First, they examined feature representation strength by plotting $\|W_i\|$ for each feature i, which indicates whether a feature is fully represented ($\|W_i\| \approx 1$) or not learned ($\|W_i\| \approx 0$). Second, to understand whether features share dimensions with other features, they calculated interference $\sum_{j \neq i} (\hat{W}_i \cdot W_j)^2$, which projects all other features onto the direction vector of feature i. The simplest way to visualize this is using the $W^T W$ matrix, which is an identity matrix for the most important features and zero for features with no interference. Positive values in the matrix express that when one feature activates, it activates the other feature partially and negative values express that when one feature activates, it inhibits the other feature partially.

In order to define monosemanticity and polysemanticity across various models as a statistical property of features the model represents, following Elhage et al. [2022] we take "dimensionality" for a feature i as:

$$D_i = \frac{\|W_i\|^2}{\sum_j (\hat{W}_i \cdot W_j)^2}$$

where W_i is the weight vector of the ith feature and $\hat{W}_i$ is the unit vector in the direction of W_i. In order to compare the effects of feature sparsity, S across the dense and MoE models we take "dimensions per feature" following Elhage et al. [2022] as

$$D^* = \frac{m}{||W||_F^2}$$

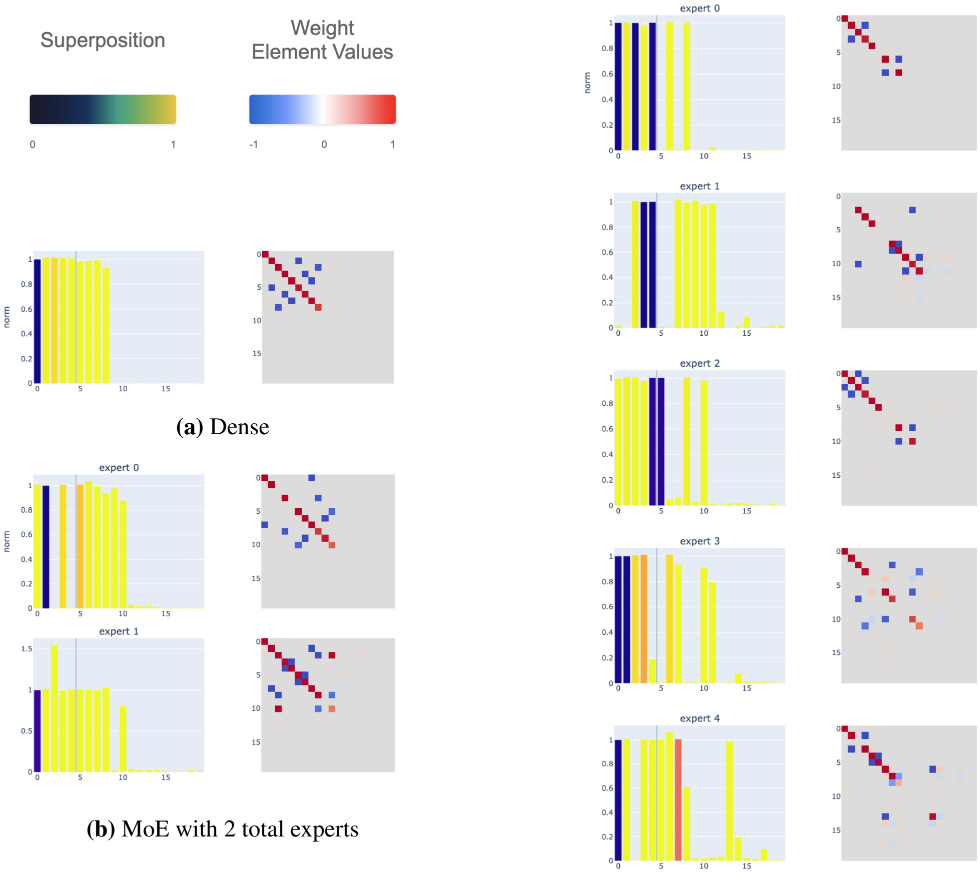
where m is the number of hidden dimensions and $||W||_F^2$ is the Frobenius norm of the weight matrix.

3 Demonstrating Superposition

In order to understand the phenomenon of superposition in toy MoEs, we measured the norm of a feature weight vector in an expert e given by $\|W_i^{(e)}\|$. It represents the extent to which a feature is represented within the expert e. $\|W_i^{(e)}\| \approx 1$ if the feature i is fully represented in expert e and zero if it is not learned. Furthermore, we extended the W^TW matrix per expert to an MoE.

We varied the total number of experts, E, keeping all other parameters constant, including: number of features n, hidden dimensions m, feature density $1 - S$, and feature importance I of each feature. The goal was to isolate the effect of network sparsity (k/E) on superposition. We compared $\|W_i^{(e)}\|$ and W^TW for each expert e with the dense model for $E = 2$ and $E = 5$.

We observed that each expert e in a MoE represents more features monosemantically (less superposition) than the dense model, signified by the greater number of purple bars in Figure 1b and 1c. Increasing the total number of experts E from 2 to 5 further increases the number of monosemantic features per expert. The features in each expert in these MoEs also have relatively less interference with every other feature compared to the dense model.



(a) Dense

(b) MoE with 2 total experts

(c) MoE with 5 total experts

Figure 1: Demonstration of superposition in (a) Dense, (b) MoE with 2 total experts, and (c) MoE with 5 total experts for $n = 20$, $m = 5$, $I = 0.7^i$ and $1 - S = 0.1$. The left figure in each pair represents the norm of each feature's direction vector $\|W_i\|$. Each feature's color represents whether the feature is orthogonal to other features (i.e. in superposition). The right figure in each pair represents the W^TW matrix and each cell is colored by the dot product between feature weight vectors. For MoEs (b) and (c), the visualizations are shown per expert.

We extend the feature dimensionality to be per expert by defining expert-specific feature dimensionality as $D_i^{(e)} = \frac{\left\| W_i^{(e)} \right\|^2}{\sum_j \left(\hat{W}_i^{(e)} \cdot W_j^{(e)} \right)^2}$ where $\left\| W_i^{(e)} \right\|^2$ is the squared L2 norm of the weight vector of the ith feature in expert e and $\hat{W}_i^{(e)}$ is the unit vector in the direction of $W_i^{(e)}$. It represents the "fraction of a dimension" that a specific feature gets. Features that are monosemantic in an expert will have a dimensionality of one while features that are not learned by an expert will have dimensionality zero.

To directly compare dimensionality of the features in a dense model, D_i, to the features in an MoE, we define the **global dimensionality** of a feature i in a MoE in terms of expert-specific feature dimensionality weighted by the activation rate of the expert given that feature i is present:

$$D_i^{\text{global}} = \sum_e \alpha_i^{(e)} \cdot D_i^{(e)}$$

where D_i^e is the expert-specific feature dimensionality and $\alpha_i^{(e)} = P(\text{expert } e \text{ active} \mid \text{feature } i \text{ present})$. $\alpha_i^{(e)}$ represents the number of times expert e is active when feature i is present divided by the total number of times feature i is present across a batch of input samples.

We also compute per-expert dimensions per feature, $D^{*(e)} = \frac{m}{||W^{(e)}||_F^2}$ and a global dimensions per feature for a MoE, D^*_{global} defined as:

$$D^*_{global} = \frac{E \cdot m}{||W_{combined}||_F^2}$$

where $W_{combined}$ is the stacked weight matrix across E experts, $[W^{(1)}; W^{(2)}; \ldots; W^{(E)}]$. To understand how differently features in a MoE occupy space compared to the dense models, we compute D^*_{global} at varying sparsity levels.

Concretely, we observed the global dimensionality of each feature i in an MoE is almost always **greater** than the dimensionality of that feature in the dense model as shown in Figure 2. Higher global dimensionality means that an MoE allocates larger fraction of the dimension to the same feature across various models. Furthermore, across all sparsity values, MoEs with 5 and 8 experts have a higher number of hidden dimensions per feature than the dense models as shown in Figure 3. Higher D_i^{global} and D^*_{global} for the MoEs compared to the dense model signifies that the features in the MoEs are less polysemantic, as they have less interference from other features (Figure 1).

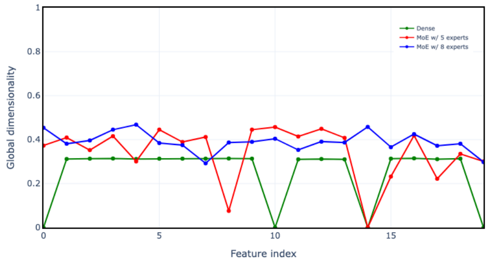


Figure 2: Global feature dimensionality, D_i^{global}, for the dense, MoE with 5 experts, and MoE with 8 experts for $n = 20, m = 5, I = 1.0$.

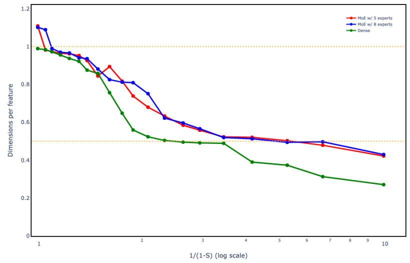


Figure 3: Number of hidden dimensions per embedded feature across the dense, MoE with 5 experts, and MoE with 8 experts for $n = 20, m = 5, I = 1.0$ and varying log sparsity $1/(1 - S)$.

4 Phase Change

Dense toy models exhibit discontinuous 'phase changes' between internal feature representations [Elhage et al., 2022]. By varying the properties of the input distribution, we can elicit different behavior. In general, more feature sparsity encourages greater superposition. Analyzing how

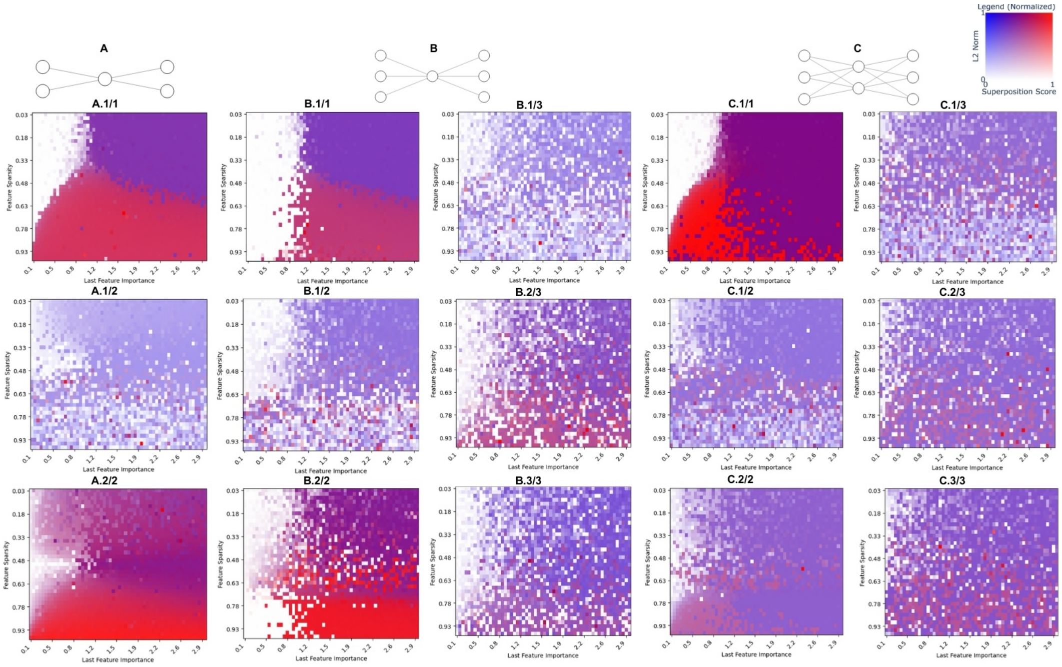


Figure 4: Joint feature norm ($||W_i||^2$) and superposition score ($\sum_{j\neq i}(\hat{W}_i \cdot W_j)^2$) across varying feature sparsity $S \in [0.1, 1]$ and relative last feature importance $r \in [0.1, 3]$. For each cell, we train ten models and select the one with the lowest loss. We used load balancing loss in this section. We plot joint feature norm and superposition for the last feature: low L2 norm ($||W_i||$) is white, denoting the model is ignoring the last feature; otherwise a low superposition score is blue-purple to indicate monosemantic representation of the last feature and red for a high superposition score. Subfigure X.e/E denotes the weight matrix of expert e of E total experts trained on architecture X; X.1/1 is a dense model.

MoE phase diagrams change as a function of network sparsity enables a direct characterization of superposition within models and offers insight into how experts specialize.

We report the expert-specific phase diagram across all-feature sparsity and last-feature relative importance for varying network sparsity by increasing the experts (E) up to the number of input feature dimensions (n).

In all single-expert (dense) cases, we observed a clear phase change (Figure 4.X.1/1), affirming the work of Elhage et al. [2022]. When we increased the total number of experts, discrete phase changes disappeared. Some experts in MoEs with $E = 2$ are reminiscent of their respective dense cases (Figure 4.X.2/2), but exhibit more continuous transitions. In each case, the first expert became more monosemantic, specializing in the most important feature by relative importance. Experts dissimilar from the dense cases universally have much lower superposition scores (they are bluer), indicating more monosemantic representations. This aligns with the conclusions of the previous section—MoEs favor lower superposition scores compared to their dense counterparts.

For the $n = 2, m = 1$ setup (Figure 4.A), the dense model does not represent the last feature when feature sparsity is low. However, the comparable MoE model preserves the last feature much more because it has the capacity. With three input dimensions (Figure 4.B), the MoE does not exhibit this behavior because the experts are superimposing the other two features; there is no space for the third feature within one hidden dimension. For $m = 2$ the white region in the dense model (Figure 4.C.1/1) (in the mid- to low-feature sparsity domain when the feature is relatively less importance than the others) ignores the last feature. However, as network sparsity increases—across all other Figure 4.C—the models allocate the last feature greater L2 magnitude ($||W_3^1|| < ||W_3^2|| < ||W_3^3||$), choosing to represent it monosemantically instead. In other words, the dimensionality in the low relative-importance region increased with increasing network sparsity, as demonstrated in Figure 3.

We observed a window of feature sparsity from roughly 0.48 to 0.7 for Figures 4.B.2/2, 4.C.1/2, and 4.C.2/2 where there is heavy mix of polysemanticity, monosemanticity, or ignorance. This indicates there is a middleground in MoEs with comparable loss between polysemantic and monosemantic representations which make it difficult to consistently commit to the strategies we observe in low and high feature sparsity domains.

In general, MoEs have much lower superposition scores than their dense counterparts. Furthermore, an expert specializing over a subset of the feature space, as determined by the router, rarely resembles an expert specializing over the whole feature space—even when one only expert is active at a time.

5 Expert Specialization

The definition of expert specialization in MoEs traditionally centers around load balancing between experts across all inputs [Chaudhari et al., 2025]. However, this definition fails to capture the natural intuition of specialization, wherein an expert is only used when appropriate concepts—those the expert is specialized in—are present in the input.

We define an expert to be specialized if it occupies certain feature directions in the input space, and if it represents said features relatively monosemantically. We demonstrate that these two conditions are directly correlated, and show how the presence of these two conditions encourages load balancing across experts. Furthermore, we demonstrate that this understanding of specialization suggests gate initialization schemes which improve performance and encourage load balancing. We elaborate on a related, motivating idea—there exists a formulation of a MoE which monosemantically encodes a polysemantic dense model in a high sparsity domain—in Appendix A.2.

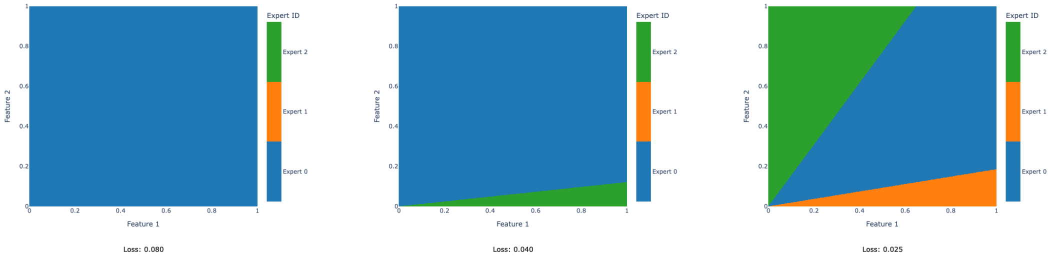
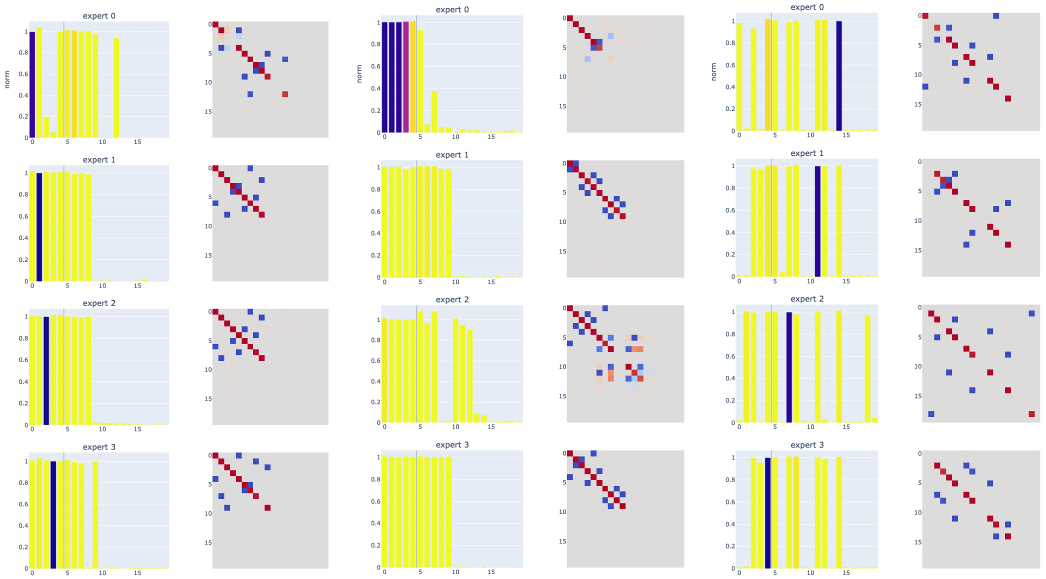


Figure 5: Expert routing of three identical models with differing initialization schemes. We use $n = 2$, $E = 3$, $m = 1$. The first model (left) has the worst performance (loss: 0.08) and routes all inputs to one expert. The second model (middle) has better performance (loss: 0.04) and routes a small portion of inputs, specifically those when feature 1 is active, to a second expert. The third model (right) has the lowest loss (loss: 0.025), and distributes the input space among all experts. One expert is chosen when only feature 1 is active, one when only feature 2 is active, and one when both are active.

There is a correlation between the distribution of experts across the feature space and how well a model performs, as shown in Figure 5. Because the router function is linear, experts occupy certain directions in the feature space, where scalar multiples of those directions are consistently routed to a single expert. Not only do models with better distribution of experts across the feature space perform better, but they tend to align experts with particular features. This implies that experts *can* specialize to specific features, and that doing so improves performance.

Next, we explore whether initializing experts over specific features in the input space cause them to be more monosemantic w.r.t. those features. Then, we see if, for the features an expert has chosen to represent monosemantically, the expert is chosen more often when that feature is active in the input.

When the gate matrix is initialized to the diagonal, such that inputs with feature i active are routed to expert i, each expert monosemantically represents the single feature it was initially aligned with, and only that feature, as shown in Figure 6a. When the router is ordered k-hot initialized, the first expert monosemantically represents four of the five features it was initialized with, as shown in Figure 6b. The other experts, initialized over other features, did not monosemantically represent these less important features, nor did they monosemantically represent the five most important features they were not initialized over. When we break the ordering of feature importance and randomize the



(a) Diagonal Initialization **(b)** Ordered K-hot initialization **(c)** Random K-hot initialization

Figure 6: $||W_i^{(e)}||^2$ and W^TW results for three different initialization schemes, with $n = 20, m = 5, E = 4, S = 0.1$. In **(a)**, the gate matrix is initialized along the diagonal, and relative feature importance decreases exponentially in order from feature one to 20. In **(b)**, the gate matrix is initialized to an "ordered k-hot", such that the first expert aligns with the first five features, and each subsequent expert aligns with the next five features. Relative feature importance is the same as **(a)**. In **(c)**, the gate matrix is initialized to a "random k-hot", where each expert is assigned five random features such that experts share no common feature but cover all 20 features collectively. Relative feature importance decreases exponentially but is randomly distributed across features.

features each expert aligns with, each expert monosemantically represented only the most important feature it was initialized over, as shown in Figure 6c.

There is a strong correlation between the features that initially are routed to an expert and which features that expert represents monosemantically. Furthermore, we observe that experts only specialize on important features—or they don't specialize at all. This is true if we give each expert one important feature explicitly, or if we give it a set of features, upon which it selects the most important feature itself and gives that one a dedicated dimension in the activation space ($D_i^{\text{global}} = 1$).

We investigated whether there is a correlation between experts representing certain features monosemantically, and said experts being chosen when those features are active in the input. We measure which features an expert monosemantically represents, then create synthetic batches where said features are active. The correlation holds both in xavier and k-hot initialization schemes, as seen in Table 1. Given $E = 10$, a mean expert usage of $\sim$10% indicates an even load balancing across experts. In all cases, when the corresponding monosemantic feature(s) for an expert is active, the usage of the expert increases significantly. When this feature(s) is the only active feature, the expert dominates the usage.

We take this increased usage to be equivalent to saying "the expert aligns with the direction of this combination of features in the input space" – when these features are active, the input vector points in the direction of the specialized features, and the expert is chosen more often.

As experts represent more features monosemantically, they can be seen as more specialized. Their usage on arbitrary input decreases, but conditional on their specialized features being active, their usage increases far greater than other experts. This holds true for all cases except the xavier initialized model with a four monosemantic feature expert, where there is a significant drop in utilization. However, when these four features are the only features active, said expert is chosen 100% of the time.

Table 1: Monosemantic feature and usage statistics per expert for $n = 100, m = 10, E = 10$. One hundred models are trained for each initialization scheme (xavier and k-hot), providing 1000 experts in total for each. Each statistic is aggregated across models, classifying experts based on the number of features they represent monosemantically. For the feature(s) an expert represents monosemantically, we track the expert usage when said feature(s) is one of several active features in the input, as well as the expert usage when said feature(s) is the *only* active feature in the input.

Xavier Initialization

Number if monosemantic features per expert	Number of experts (out of 1000)	Mean expert usage (%)	Mean expert usage; feature(s) active (%)	Mean expert usage; only feature(s) active (%)
0	461	–	–	–
1	387	9.595	17.94	67.18
2	138	9.599	30.29	95.65
3	13	8.363	40.19	100.0
4	1	1.428	14.69	100.0
5	0	–	–	–

K-Hot Initialization

Num monosemantic features per expert	Num experts (out of 1000)	Mean expert usage (%)	Mean expert usage feature(s) active (%)	Mean expert usage only feature(s) active (%)
0	335	–	–	–
1	382	10.00	23.94	100.0
2	227	10.02	46.61	100.0
3	47	10.09	62.00	100.0
4	8	9.95	70.30	100.0
5	1	9.62	74.79	100.0

The k-hot initialization makes experts more *specialized*, as shown in Table 1. There are more experts which represent features monosemantically, and there is a much stronger correlation of the router choosing an expert when its combination of features is active in the input.

Furthermore, load balancing between experts is greater for k-hot initialization compared to xavier, with a lower standard deviation of usage and higher median usage, as shown in Table 2. The k-hot initialization is able to represent more features and has higher D_i^{global}.

Table 2: Specialization and Representation Statistics Across Initialization Schemes

Initialization Scheme	Average median usage of experts	Average std of usage of experts	Average num features represented	Average global dimensionality
Xavier	0.08621	0.01171	28.730	0.1110
K-Hot	0.09956	0.00079	32.820	0.1140

6 Limitations

Our findings are based on simple autoencoder toy models with synthetic data. The extent to which these results generalize to large-scale transformer architectures with complex, high-dimensional representations remains an open question. We studied only top-$k = 1$ routing with simple linear gates. Large MoE systems employ more sophisticated routing mechanisms, multiple active experts, and architectural complexities not captured in our framework. Our analysis assumes features are independent with known sparsity patterns. Real data likely exhibit complex feature correlations and unknown sparsity structures that may fundamentally alter superposition dynamics.

7 Conclusion

We explored how network sparsity impacts superposition in MoEs. Superposition is a foundational concept of mechanistic interpretability. MoEs are used in state-of-the-art language models. For mechanistic interpretability to be useful, we must develop an understanding of superposition in these architectures.

We expanded metrics for quantifying superposition to MoEs and demonstrated they exhibit greater monosemanticity relative to their dense counterparts. We demonstrated experts in MoEs do not exhibit discrete phase changes like dense models. Finally, we offered a notion of MoE specialization not motivated by load balancing but by in terms of features. We show we can force this specialization with weight initialization schemes in place of a load balancing loss and that such models represent more features.

MoEs are commonly regarded as a method for scaling model size, less for their performance increases. However as Shazeer et al. [2017], Dikkala et al. [2023], Li et al. [2025] have demonstrated, MoEs achieve lower loss for similar active parameters or FLOPs. We observed similar patterns across all our toy models and visualized the loss for select phase change models in Appendix A.1.

Toy MoEs achieve lower loss compared to dense models—effectively trading increased network sparsity for more interpretable, less-superimposed, better-performing features. This is at odds with the general zeitgeist—that mechanistic interpretability and performance are orthogonal objectives. Of course, these are toy models. We leave future work to determine how well these ideas scale.

References

Marmik Chaudhari, Idhant Gulati, Nishkal Hundia, Pranav Karra, and Shivam Raval. Moe lens - an expert is all you need. In *Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*, 2025. URL `https://openreview.net/forum?id=GS4WXncwSF`.

Nishanth Dikkala, Nikhil Ghosh, Raghu Meka, Rina Panigrahy, Nikhil Vyas, and Xin Wang. On the benefits of learning to route in mixture-of-experts models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9376–9396, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.583. URL `https://aclanthology.org/2023.emnlp-main.583`.

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. *Transformer Circuits Pub*, 2022. URL `https://transformer-circuits.pub/2022/toy_model/index.html`.

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`.

Google DeepMind. Gemini 2.5 pro. `https://deepmind.google/technologies/gemini/pro/`, 2025.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Houyi Li, Ka Man Lo, Ziqi Wang, Zili Wang, Wenzhen Zheng, Shuigeng Zhou, Xiangyu Zhang, and Daxin Jiang. Can mixture-of-experts surpass dense llms under strictly equal resources?, 2025. URL `https://arxiv.org/abs/2506.12119`.

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`.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Appendix

A.1 Phase Change Loss

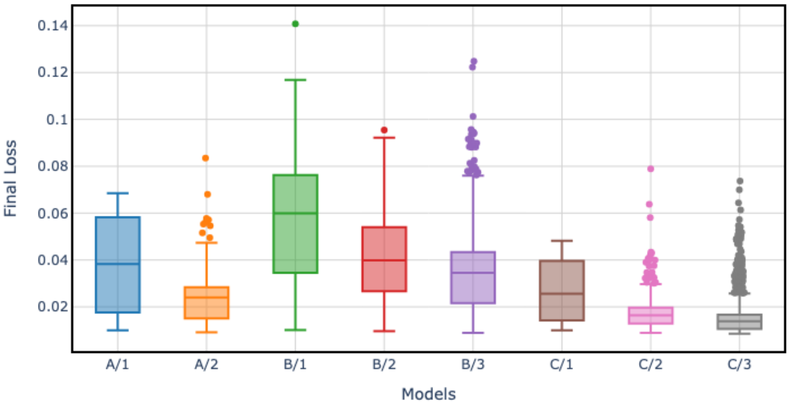


Figure 7: Model X/E uses X to denote the same model architectures and models used in Figure 4 and E denotes the total number of experts (e.g. network sparsity). Increasing network sparsity decreases mean loss while increasing localized variance—especially as the number of experts reaches the input feature dimensions. This can attributed to the relatively unstable training of MoEs compared to dense models (despite training ten models for each cell and selecting the lowest loss).

A.2 Analytic Model Equivariance

For the toy setup of single-layer, single-nonlinearity, top-$k = 1$ MoEs, there exists a theoretical map between any dense model and a monosemantic MoE with an equivalent number of active features under a sparsity constraint.

Assume there exists an upper bound for the number of active features a for any input such that $\forall x \in D : |\{i : x_i \neq 0\}| \leq a$. Furthermore, assume that a is no greater than the hidden dimensionality, m, of an expert, providing an upper bound on the number of features a model has to represent. Assume also that the hidden dimensions is smaller than the total number of input features n ($a \leq m \leq n$). To construct the monosemantic MoE, for each possible subset $S \subseteq \{1, 2, \ldots, n\}$ with $|S| \leq a$—meaning the size of the subset of active features is smaller than or equal to a—create an expert which monosemantically preserves those features. (In fact, you can take only the subsets such that $|S| = a$.) The router then selects the expert which corresponds to those active features (of which there will never be more than a, by assumption):

$$\text{Router}(x) = \arg\max_S \mathbb{I}[\text{support}(x) = S]$$

where $\text{support}(x) = \{i : x_i \neq 0\}$. Since $|S| \leq a \leq m$, each expert has sufficient capacity to represent its assigned features without superposition. To reiterate, only a features are active and every unique combination of active features receives its own dedicated expert with sufficient capacity to represent those features monosemantically. So, the number of possible experts needed is $\binom{n}{m}$.

The reconstruction for this theoretical MoE has zero loss only as the sparsity constraint holds (or goes to one in these toy models) because there is the chance more than m features could be active at one time ($a \not\leq m$), which would exceed the monosemantic representational capacity of the network (but the dense polysemantic could do no better unless features are correlated in the distribution). Therefore, even if $a \not\leq m$ sometimes, the polysemantic model encounters the same problem and the monosemantic MoE under this construction may still outperform it under looser sparsity constraints.

Thus, for any dense model, $f_{\text{dense}}(x) = \text{ReLU}(Wx + b)$ under the sparsity constraint $|\text{support}(x)| \leq a$, there exists an MoE model $f_{MoE}(x)$ such that $f_{\text{dense}}(x) = f_{\text{MoE}}(x)$ for all valid inputs. In the toy settings described in this paper, the sparsity constraint holds in the limit where sparsity goes to one. However, in practice there may be an upper bound on the amount of features a particular amount of information can semantically encode, indicated by the size of meaningful embeddings of that data. Therefore, a MoE model with sufficient experts and a tractable amount of superposition (e.g. interpretable) may be sufficient to encode all features present.