Understanding sparse autoencoder scaling in the presence of feature manifolds

Anonymous Author(s)
Affiliation
Address
email

Abstract

Sparse autoencoders (SAEs) model the activations of a neural network as linear combinations of sparsely occurring directions of variation (latents). The ability of SAEs to reconstruct activations follows scaling laws w.r.t. the number of latents. In this work, we adapt a capacity-allocation model from the neural scaling literature (Brill, 2024) to understand SAE scaling, and in particular, to understand how *feature manifolds* (multi-dimensional features) influence scaling behavior. Consistent with prior work, the model recovers distinct scaling regimes. Notably, in one regime, feature manifolds have the pathological effect of causing SAEs to learn far fewer features in data than there are latents in the SAE. We provide some preliminary discussion on whether or not SAEs are in this pathological regime in the wild.

1 Introduction

Sparse autoencoders [1–9] and related methods [10–14] decompose neural network activations into a collection of sparsely activating latents. As SAEs have been scaled (now to millions of latents) [4, 15–17], they have exhibited *scaling laws* [4, 15, 18], where loss improves predictably as a power law with the number of latents in the SAE. Although sparse autoencoders learn many interpretable features, some worry that they may miss important structure in neural representations, either because there are an extraordinary number of very rare features in activations [19] or because the SAE architecture and training objective make incorrect assumptions about the structure of neural representations [20, 21].

In this work, we develop a formal analysis of SAE scaling behavior and scaling laws. We are particularly interested in understanding SAE scaling when activations contain a particular kind of structure: *feature manifolds* (multi-dimensional features) [22, 23], and in whether feature manifolds could cause pathological scaling and exacerbate the problem of interpretability "dark matter" [19]. Our main approach is to adapt a mathematical model of *neural scaling* from Brill (2024) [24], where models allocate capacity between different data manifolds, to the case of SAEs. Guided by our model, we then conduct experiments to probe whether SAEs may scale pathologically in practice.

2 A model of sparse autoencoder scaling

2.1 The structure of data and the SAE architecture

Activation data: We assume the multi-dimensional linear representation hypothesis [25, 26] and that neural network activation vectors $\mathbf{x} \in \mathbb{R}^d$ are generated as a sum of sparsely occurring *features*: $\mathbf{x} = \sum_i \mathbf{S}_i \mathbf{f}_i$ where $\mathbf{S}_i$ is the subspace where feature i lives, specified by a $d \times d_i$ matrix whose columns are basis vectors of the subspace, and $\mathbf{f}_i$ is a random variable taking values in $\mathbb{R}^{d_i}$, where d_i is the *dimension* of feature i. Each feature $\mathbf{f}_i$ is *sparse*, supported on a small fraction of the data: $p_i = \Pr[\mathbf{f}_i \neq \mathbf{0}] \ll 1$, so each activation vector is a sum of only a small subset of all features.

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

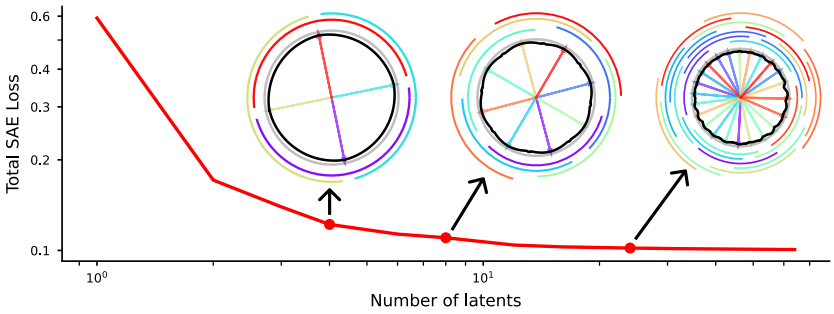

Figure 1: **SAE scaling on a toy feature manifold** S^1. We train ReLU SAEs with an L1 sparsity penalty to reconstruct points on the circle $S^1 \subseteq \mathbb{R}^2$. We find that SAEs can slightly reduce their total loss by "tiling" the manifold with more sparsely activating latents. For SAEs with 4, 8, and 24 latents, we show the data (S^1) in grey, the SAE's reconstruction in black, the decoder latent directions as arrows, and indicate the part of the circle that each latent fires on as colored arcs. If SAEs can reduce loss by "tiling" a manifold in this way, they may do this at the expense of learning rarer features.

SAEs: Sparse autoencoders attempt to reconstruct activation vectors as a sum of sparsely activating latents, consisting of an encoding step $\hat{\mathbf{f}} = \text{Enc}(\mathbf{x}) = \sigma(\mathbf{W}_e\mathbf{x} + \mathbf{b}_e)$ where σ is a nonlinearity and $\hat{\mathbf{f}} \in \mathbb{R}^N$, and a decoding step $\hat{\mathbf{x}} = \text{Dec}(\hat{\mathbf{f}}) = \mathbf{W}_d\hat{\mathbf{f}} + \mathbf{b}_d$. Sparse autoencoders are trained with SGD on a loss $\mathcal{L} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \lambda\, S(\hat{\mathbf{f}})$ where $S(\hat{\mathbf{f}})$ is a sparsity-encouraging loss like $\|\hat{\mathbf{f}}\|_1$ or $\|\hat{\mathbf{f}}\|_0$.

2.2 Assumption: SAE optimization reduces to a latent allocation problem

We assume that SAEs learn solutions where each latent j is *specific* to a particular feature i, so that $\hat{\mathbf{f}}_j \neq 0$ only if $\mathbf{f}_i \neq \mathbf{0}$. We further assume that the SAE latent decoder directions for the latents associated with feature i lie within the subspace $\mathbf{S}_i$, and that the subspaces where features live are orthogonal, i.e., $\mathbf{S}_i \perp \mathbf{S}_k$ if $i \neq k$. It follows that the SAE loss on any sample $\mathbf{x}$ is the sum of losses on samples where each active feature had fired alone: if $\mathcal{A}(\mathbf{x})$ are the "active" features i where $\mathbf{f}_i \neq \mathbf{0}$ on $\mathbf{x}$, then

$$\mathcal{L}(\mathbf{x}) = \mathcal{L}\left(\sum_{i\in\mathcal{A}(\mathbf{x})} \mathbf{S}_i\mathbf{f}_i\right) \overset{\text{assumption}}{=} \sum_{i\in\mathcal{A}(\mathbf{x})} \mathcal{L}(\mathbf{S}_i\mathbf{f}_i).$$

We note that feature absorption and related phenomena [27] and the non-orthogonality of features in practice violate these assumptions, but we will accept them for the sake of expedience. If SAE latents are specific to features, then an optimal SAE's loss on a feature i will be determined by the number of latents n_i the SAE allocates to reconstructing feature i, which we denote $L_i(n_i)$, and will be determined by the geometry of the feature $\mathbf{f}_i$. The expected loss across activations $\mathbf{x}$ is then:

$$\mathbf{E}_\mathbf{x}\left[\mathcal{L}(\mathbf{x})\right] = \mathbf{E}_\mathbf{x}\left[\sum_{i\in\mathcal{A}(\mathbf{x})} \mathcal{L}(\mathbf{S}_i\mathbf{f}_i)\right] = \sum_i p_i\mathcal{L}(\mathbf{S}_i\mathbf{f}_i) = \boxed{\sum_i p_i L_i(n_i)}$$

We therefore reduce the SAE optimization problem to the problem of choosing how many latents n_i to allocate to each feature i in the data. To understand SAE scaling behavior, we solve for the allocation of latents to each feature n_i that minimizes $\sum_i p_i L_i(n_i)$ given a distribution over feature frequencies p_i, the per-feature loss curves L_i, and the constraint on the total number of latents $\sum_i n_i = N$.

2.3 Warm-up on discrete features

To get comfortable with this formulation, we first apply it to the simplest case where all features are *discrete*. We imagine that for all features i, $d_i = 1$ and $\mathbf{f}_i = 1$ on the fraction p_i of samples where feature i is present and is 0 otherwise. For each feature i, an SAE incurs loss 1 if $n_i = 0$ (reconstruction loss = 1 and sparsity loss = 0). If $n_i = 1$, it can perfectly reconstruct the feature, incurring sparsity loss λ with L0 or L1 sparsity loss. Therefore $L(n) = 1$ if $n_i < 1$ else λ.

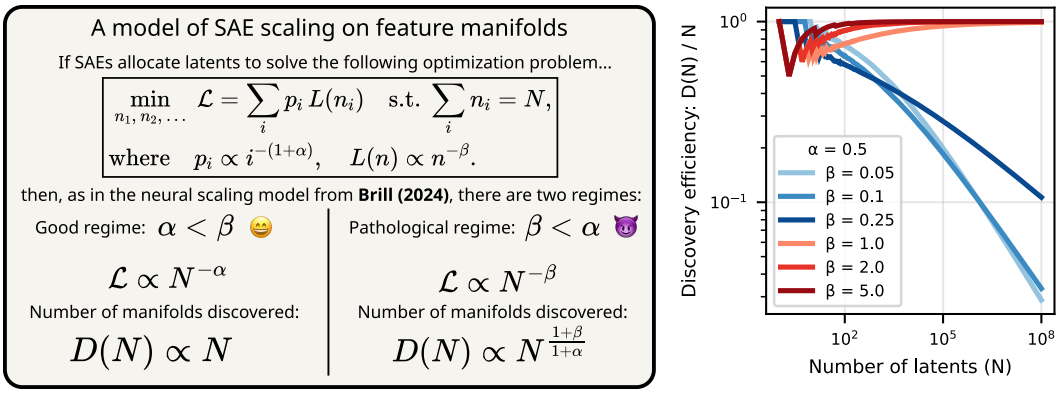

Figure 2: **Left**: Application of Brill's (2024) [24] capacity-allocation model to SAE scaling. **Right**: numerical simulation of SAE scaling when the most frequently occurring feature is a manifold with loss scaling as $L(n) \propto n^{-\beta}$ and all other features are discrete. We see that if $\beta \ll \alpha$, then a simulated SAE with 100 million latents discovers only 3 million features.

With total loss $\sum_i p_i L(n_i)$ and N latents in our SAE we see that the optimal solution is to allocate one latent to the most commonly occurring N features. In this setup, then, **the effect of scaling SAEs is to learn an increasing number of discrete features in the data, in decreasing order of their frequency**. We will often be interested in the number of features "discovered" by the SAE $D(N) = |\{i \ : \ n_i > 0\}|$. In this setting, $D(N) = N$. To recover the power law scaling that SAEs exhibit empirically, we only need to assume that $p_i \propto i^{-(1+\alpha)}$. With this assumption, the improvement in total loss from adding a marginal latent i follows $p_i(1-\lambda)$, and integrating from $i = 1 \ldots N$ we get that the total loss drops off as a power law $\mathcal{L}(N) \propto N^{-\alpha}$. We note that this model of SAE scaling mirrors the "quanta" model of neural scaling from [28], where here the "quanta" are features. This picture also agrees with the finding from [15] that SAEs learn features for concepts in data approximately in order of the frequencies (roughly Zipfian) at which those concepts occur.

2.4 Intuition behind pathological manifold scaling

Recently, several works have commented on the existence of *feature manifolds* [12, 23] (*multi-dimensional features*) [22] in neural networks. In our formalism above, these are features i with $d_i > 1$ and where the range of $\mathbf{f}_i$ is a manifold embedded in $\mathbb{R}^{d_i}$. With feature manifolds, instead of having a discrete $L_i(n_i)$ curve like above, L_i might drop off slowly as n_i grows. To show that this is possible, in Figure 1, we show the scaling curve for ReLU SAEs ($\sigma = \text{ReLU}$) trained with L1 penalty to reconstruct points on a circle $\mathbf{x} \in S^1 \subseteq \mathbb{R}^2$. We observe that these SAEs can gradually reduce their total loss by more finely "tiling" the manifold with latents that activate more sparsely, and that this manifold can accommodate dozens of latents before loss plateaus. While such solutions ruthlessly minimize the SAE loss, it is not obvious that they would be better from an interpretability standpoint. Our fundamental concern is that if $L_i(n_i)$ curves decrease gradually, it could be optimal for an SAE to tile common feature manifolds instead of discovering rarer features in data.

2.5 Solution for power-law L(n) following Brill (2024)

We assume that feature frequencies decay as a power law $p_i \propto i^{-(1+\alpha)}$ and that all features have the same power law per-feature loss curve $L_i(n_i) = n_i^{-\beta}$. In this setting, our model of SAE scaling directly corresponds to the model of neural scaling from Brill (2024) [24], where neural networks allocate units of capacity n_i towards approximating functions on distinct power-law distributed data manifolds, with per-manifold loss scaling as $n_i^{-c/D}$. We show similar derivations to Brill [24] in Appendix B and summarize the core results in Figure 2 (**left**).

In our notation, there is a key threshold: $\alpha < \beta$. When $\alpha < \beta$, $D(N) \propto N$, so the "efficiency" at which SAEs discover features $D(N)/N$ tends towards a constant. However, when $\beta < \alpha$, then $D(N) \propto N^{\frac{1+\beta}{1+\alpha}}$, and so $D(N)/N$ tends towards 0. This illustrates the core dynamic of concern:

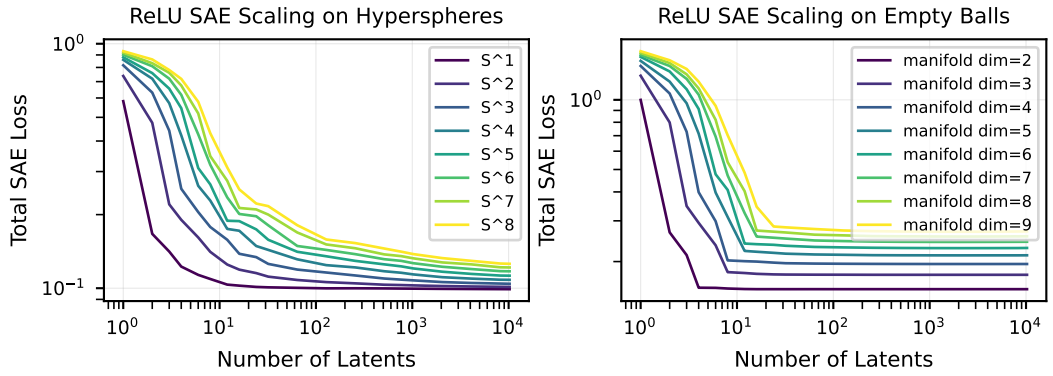

Figure 3: ReLU SAE scaling on individual toy feature manifolds, showing how $L(n)$ curves depend on feature geometry. We train with L1 sparsity $\lambda = 0.1$. **Left**: SAE scaling on unit hyperspheres of varying dimension. **Right**: SAE scaling on points sampled in $\{\mathbf{x} : 0.5 < |\mathbf{x}| < 2\}$.

when SAEs can continue to reduce loss by "tiling" common feature manifolds, then it can be optimal for them to do this at the expense of discovering other, rarer features.

2.6 Numerical simulation

While our mathematical model above assumed that all features have the same $L_i(n_i)$ curve, we can run numerical simulations with arbitrary $L_i(n_i)$ curves. In Figure 2 (**right**), we conduct simulations where the first feature scales as $n^{-\beta}$ but all other features are discrete and scale as step functions $\mathbf{1}_{n=0}$, and with $p_i \propto i^{-(1+0.5)}$. We find that, with only a single feature with power-law $L(n_i)$ scaling, when $\beta < \alpha$ this feature begins to absorb the vast majority of latents in the SAE once N is large.

3 SAE scaling on synthetic features and on real neural networks

In our analysis, whether SAEs will scale pathologically depends on the shape of their per-feature loss curves $L_i(n_i)$. We showed one such $L(n)$ curve in Figure 1, but we further explore this by training SAEs on a variety of synthetic manifolds in Figure 3 and in Appendix Figure 6. Overall, we find that the shape of the $L(n)$ curve depends on the manifold geometry, with some manifolds accommodating many thousands of latents without saturating loss while on others the loss curve plateaus very quickly (effectively $\beta \to \infty$). In Appendix A.1, we provide further discussion on what α and β may be, and whether SAEs might be scaling pathologically, in practice.

When a large number of SAE latents are allocated to a relatively low-dimensional feature manifold, we'd expect the cosine similarities between the decoder latents allocated to that manifold to be close to 1. In Appendix E, we show the distribution over decoder latent nearest neighbor cosine similarities for SAEs trained on LLM and vision model activations, and find some differences between them.

4 Discussion

In this short paper, we have described a model of SAE scaling which reduces the SAE optimization problem to the problem of optimally allocating different numbers of SAE latents to different features in data. As in the model of scaling from Brill [24] SAE scaling laws either result from an underlying power law distribution over features $p_i \propto i^{-(1+\alpha)}$, or from the improvements in loss from tiling common feature manifolds following a power law $L_i(n_i) \propto n_i^{-\beta}$. When $\beta < \alpha$, SAE latents could massively accumulate on commonly occurring feature manifolds.

Unfortunately, we do not resolve the question of whether SAEs are in this pathological scaling regime in practice. Our uncertainty is due to our knowing neither the distribution over true feature frequencies (determining α) nor the geometry of real-world neural network feature manifolds (which determines β). We give some additional commentary in Appendix A.1. We view this work as primarily about framing, rather than completely answering, this interesting problem.

References

[1] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011.

[2] 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.

[3] Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*, 2024.

[4] 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. *arXiv preprint arXiv:2406.04093*, 2024.

[5] Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

[6] Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024.

[7] Thomas Fel, Ekdeep Singh Lubana, Jacob S Prince, Matthew Kowal, Victor Boutin, Isabel Papadimitriou, Binxu Wang, Martin Wattenberg, Demba Ba, and Talia Konkle. Archetypal sae: Adaptive and stable dictionary learning for concept extraction in large vision models. *arXiv preprint arXiv:2502.12892*, 2025.

[8] Valérie Costa, Thomas Fel, Ekdeep Singh Lubana, Bahareh Tolooshams, and Demba Ba. From flat to hierarchical: Extracting sparse representations with matching pursuit. *arXiv preprint arXiv:2506.03093*, 2025.

[9] Mark Muchane, Sean Richardson, Kiho Park, and Victor Veitch. Incorporating hierarchical semantics in sparse autoencoder architectures. *arXiv preprint arXiv:2506.01197*, 2025.

[10] Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. Transcoders find interpretable llm feature circuits. *Advances in Neural Information Processing Systems*, 37:24375–24410, 2024.

[11] Jack Lindsey, Adly Templeton, Jonathan Marcus, Thomas Conerly, Joshua Batson, and Christopher Olah. Sparse crosscoders for cross-layer features and model diffing, October 2024. URL `https://transformer-circuits.pub/2024/crosscoders/index.html`. Research update.

[12] Liv Gorton. Group crosscoders for mechanistic analysis of symmetry. *arXiv preprint arXiv:2410.24184*, 2024.

[13] Julian Minder, Clément Dumas, Caden Juang, Bilal Chugtai, and Neel Nanda. Robustly identifying concepts introduced during chat fine-tuning using crosscoders. *arXiv preprint arXiv:2504.02922*, 2025.

[14] Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 2025. URL `https://transformer-circuits.pub/2025/attribution-graphs/methods.html`.

[15] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

[16] Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

[17] Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

[18] Jack Lindsey, Tom Conerly, Adly Templeton, Jonathan Marcus, and Tom Henighan. Scaling laws for dictionary learning, April 2024. URL https://transformer-circuits.pub/2024/april-update/index.html#scaling-laws. Circuits Updates — April 2024.

[19] Chris Olah and Adam Jermyn. The dark matter of neural networks? Transformer Circuits Thread, July 2024. URL https://transformer-circuits.pub/2024/july-update/index.html#dark-matter. Part of Circuits Updates - July 2024, Anthropic Interpretability Team.

[20] Róbert Csordás, Christopher Potts, Christopher D Manning, and Atticus Geiger. Recurrent neural networks learn to store and generate sequences using non-linear representations. *arXiv preprint arXiv:2408.10920*, 2024.

[21] Sai Sumedh R Hindupur, Ekdeep Singh Lubana, Thomas Fel, and Demba Ba. Projecting assumptions: The duality between sparse autoencoders and concept geometry. *arXiv preprint arXiv:2503.01822*, 2025.

[22] Joshua Engels, Eric J Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear. *arXiv preprint arXiv:2405.14860*, 2024.

[23] Alexander Modell, Patrick Rubin-Delanchy, and Nick Whiteley. The origins of representation manifolds in large language models. *arXiv preprint arXiv:2505.18235*, 2025.

[24] Ari Brill. Neural scaling laws rooted in the data distribution. *arXiv preprint arXiv:2412.07942*, 2024.

[25] 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 Thread*, 2022. https://transformer-circuits.pub/2022/toy_model/index.html.

[26] Joshua Engels, Logan Riggs, and Max Tegmark. Decomposing the dark matter of sparse autoencoders. *arXiv preprint arXiv:2410.14670*, 2024.

[27] David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, Satvik Golechha, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint arXiv:2409.14507*, 2024.

[28] Eric Michaud, Ziming Liu, Uzay Girit, and Max Tegmark. The quantization model of neural scaling. *Advances in Neural Information Processing Systems*, 36:28699–28722, 2023.

[29] Demian Till. Do sparse autoencoders find "true features"? https://www.lesswrong.com/posts/QoR8noAB3Mp2KBA4B/do-sparse-autoencoders-find-true-features, 02 2024. Accessed: 2025-08-11.

[30] Evan Anders, Clement Neo, Jason Hoelscher-Obermaier, and Jessica N. Howard. Sparse autoencoders find composed features in small toy models. https://www.lesswrong.com/posts/a5wwqza2cY3W7L9cj/sparse-autoencoders-find-composed-features-in-small-toy, 03 2024. Accessed: 2025-08-11.

[31] 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. *arXiv preprint arXiv:2001.08361*, 2020.

[32] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[33] Chris Olah. What is a linear representation? what is a multidimensional feature? Transformer Circuits Thread, July 2024. URL https://transformer-circuits.pub/2024/july-update/index.html#linear-representations. Part of Circuits Updates - July 2024, Anthropic Interpretability Team.

[34] Tom Conerly, Adly Templeton, Trenton Bricken, Jonathan Marcus, and Tom Henighan. Update on how we train saes. Transformer Circuits Thread, April 2024. URL https://transformer-circuits.pub/2024/april-update/index.html#training-saes. Part of Circuits Updates - April 2024, Anthropic Interpretability Team.

[35] Tom Conerly, Hoagy Cunningham, Adly Templeton, Jack Lindsey, Basil Hosmer, and Adam Jermyn. Dictionary learning optimization techniques. Transformer Circuits Thread, January 2025. URL https://transformer-circuits.pub/2025/january-update/index.html#DL. Part of Circuits Updates - January 2025, Anthropic Interpretability Team.

A Additional discussion

A.1 Are real SAEs in the pathological regime?

It is worth attempting to say more about whether SAEs are in the pathological scaling regime in practice. As we stated in Section 2.5, this depends on the rate at which the feature frequencies $p_i \propto i^{-(1+\alpha)}$ decay vs. the rate at which the per-feature SAE loss decays $L(n_i) \propto i^{-\beta}$. In Appendix B, following Brill (2024) [24], we show that when $\alpha < \beta$, the efficiency at which SAEs discover features $D(N)/N$ approaches a reasonable constant, but when $\beta < \alpha$, $D(N)/N$ approaches 0 as $N \to \infty$. Whether feature manifolds could cause pathological SAE scaling in the real world depends then on the real α (assuming it's even a power law) and β (assuming $L_i(n_i)$ is also a power law $\propto n_i^{-\beta}$).

What is α?: We first speculate on what α may be. One way of trying to measure this is to look at how the latent activation frequencies decay when sorted by frequency. For a few Gemma Scope SAEs [16], we show these curves in Figure 4, and measure slopes between -0.57 and -0.74. If there was a one-to-one relationship between SAE latents and features in the data, then this would imply an $\alpha \approx 0.5$ to $\alpha \approx 0.7$. However, feature absorption [27], the learning of compositional features [29, 30], and latents tiling a feature manifold could distort the relationship between the true feature frequencies and the latent activation frequencies.

Another highly speculative way of trying to estimate α could be to look at the exponents of neural scaling laws for models like those the SAE is being trained on. The idea here is that if the "features" are the computational units of neural networks that ref [28] called the "quanta", then the underlying neural scaling law slope would reflect the distribution over feature occurrences. Neural scaling exponents for language models (w.r.t. network parameters) (α_N in the scaling law $N^{-\alpha_N}$ have been measured to have an α_N 0.07 [31] and 0.34 [32], potentially implying a distribution over quanta/features $p_i \propto i^{-(1+\alpha)}$ with exponent α in that same range.

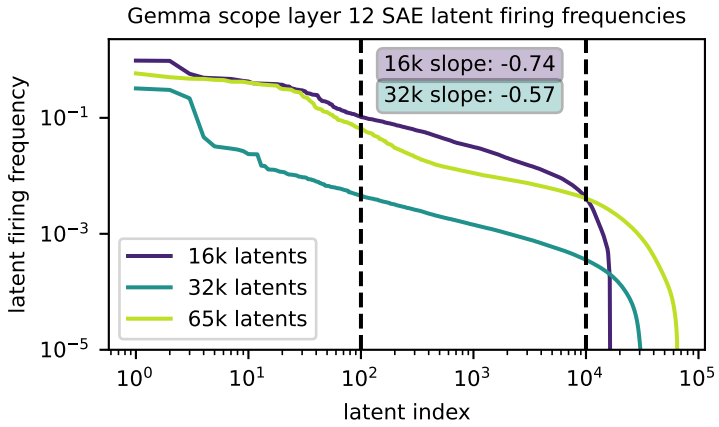

Figure 4: Frequencies at which latents fire in gemma scope SAEs, sorted by frequency. We measure the power law decay exponent between latent 10^2 and 10^4 to be -0.74 for an SAE with 16k latents and -0.57 for an SAE with 32k latents.

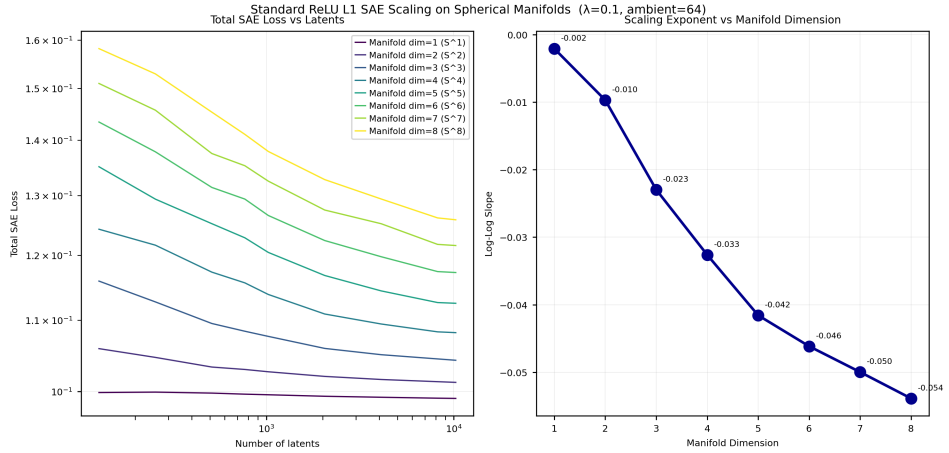

Figure 5: Measured slopes of L_i curves for ReLU L1 SAEs trained on hyperspheres.

What is β?: In Figure 3 and Figure 6, we show $L(n)$ scaling curves for SAEs reconstructing single synthetic feature manifolds. We find that these curves depend on the feature geometry. In particular, we see that when reconstructing hollow hyperspheres, we can observe gradual $L(n)$ scaling. The higher-dimensional hyperspheres in particular can accommodate at least 10^4 latents without loss plateauing. In Figure 5, we plot the slope that we measure for these curves between 10^2 and 10^4 latents, and measure a β of roughly 0.05 for hyperspheres with dimension 6-8.

However, in the more realistic setting where there is variation in the radial direction–the intensity that features fire [19]–we see that manifolds tend to saturate very quickly. It appears that the saturation happens when $n_i \approx 2d_i$, likely corresponding to solutions where the SAE latents form a basis for the subspace where the feature is embedded (or rather use two latents for each basis direction, one in the "positive" direction and one in the "negative" direction since latents can only fire positively).

Therefore, the slope of the $L_i(n_i)$ curve, and whether SAEs can use a large number of latents to reduce loss on feature manifolds, depends on the geometry of the manifold. Our experiments so far indicate that in the more realistic setting where there is variation in the radial direction (which was seen in practice in ref [22]), that SAEs do not discover solutions which take advantage of a large number of latents, and instead learn a basis solution. **This is probably the strongest argument against the possibility of feature manifolds causing pathological SAE scaling.**

Lastly, we note that we are unsure how "ripples" [33] in feature manifolds could affect SAE scaling on them. If a feature manifold is intrinsically low dimensional, but ripples through a large number of other dimensions, we could imagine SAE solutions potentially looking different.

B Derivations

B.1 Loss decomposition into per-feature terms

We work under the assumptions in §2.2: (i) *feature-specific latents*—each latent j fires only when a unique feature $i(j)$ is active; (ii) *decoder respect for subspaces*—decoder columns for latents assigned to feature i lie in $\mathrm{span}(\mathbf{S}_i)$; and (iii) *orthogonal feature subspaces*—$\mathrm{span}(\mathbf{S}_i) \perp \mathrm{span}(\mathbf{S}_k)$ for $i \neq k$.

Additivity of sparsity. For a sample $\mathbf{x} = \sum_{i \in \mathcal{A}(\mathbf{x})} \mathbf{S}_i \mathbf{f}_i$, feature-specificity implies $\hat{\mathbf{f}}_j(\mathbf{x}) = 0$ unless $i(j) \in \mathcal{A}(\mathbf{x})$. For separable sparsity penalties (L0/L1), $S(\hat{\mathbf{f}}) = \sum_j s(\hat{f}_j)$, so

$$S(\hat{\mathbf{f}}(\mathbf{x})) = \sum_{i \in \mathcal{A}(\mathbf{x})} \sum_{j:\, i(j)=i} s(\hat{f}_j(\mathbf{x}))\,.$$

Thus the sparsity cost splits across active features.

Orthogonal reconstruction. Write the model reconstruction as $\hat{\mathbf{x}} = \sum_i \hat{\mathbf{x}}_i$, with $\hat{\mathbf{x}}_i := \sum_{j:\, i(j)=i} \mathbf{w}_j \hat{f}_j \in \mathrm{span}(\mathbf{S}_i)$ by (ii). Then, using (iii),

$$\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 = \Big\| \sum_i \big(\mathbf{S}_i \mathbf{f}_i - \hat{\mathbf{x}}_i\big) \Big\|_2^2 = \sum_i \|\mathbf{S}_i \mathbf{f}_i - \hat{\mathbf{x}}_i\|_2^2\,.$$

Hence the per-sample objective $\mathcal{L}(\mathbf{x}) = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \lambda S(\hat{\mathbf{f}})$ decomposes as a sum over features active on that sample. Taking expectations and letting n_i be the number of latents allocated to feature i, we obtain

$$\boxed{\mathbb{E}_{\mathbf{x}}\big[\mathcal{L}(\mathbf{x})\big] = \sum_i p_i\, L_i(n_i), \qquad \sum_i n_i = N} \tag{1}$$

where $p_i := \Pr[\mathbf{f}_i \neq 0]$ and $L_i(n_i)$ is the (feature-i) expected reconstruction-plus-sparsity loss achieved with n_i latents restricted to $\mathrm{span}(\mathbf{S}_i)$. We note that the derivations below closely follow those in Brill (2024) [24], adapted to our SAE setting with appropriate changes in notation and interpretation. We show them here for convenience.

B.2 Scaling setup and notation

We study the optimal latent allocation n_i minimizing (1) under two empirical power-law regularities:

$$p_i \propto i^{-(1+\alpha)} \quad \text{(features sorted by frequency)}, \qquad L_i(n) \equiv L(n) \propto n^{-\beta}\,,$$

with $\alpha, \beta > 0$.[1] Define the discovery count

$$D(N) := \big|\{\, i :\ n_i > 0 \,\}\big|$$

and the total expected loss $\mathcal{L}(N) := \sum_i p_i L(n_i)$ at total width N. A standard Lagrange multiplier treatment (continuous relaxation) yields

$$n_i \ \propto\ p_i^{\frac{1}{1+\beta}} \ \propto\ i^{-\gamma}, \qquad \gamma := \frac{1+\alpha}{1+\beta}. \tag{2}$$

In practice there is a cutoff index i_c ("last discovered feature") with $n_{i_c} \approx 1$ and $n_i \lesssim 1$ for $i > i_c$. Then $D(N) \asymp i_c$ and

$$N = \sum_{i \le i_c} n_i \ \propto\ \sum_{i \le i_c} i^{-\gamma}.$$

Two regimes follow depending on whether the allocation tail-sum diverges or converges.

[1] Constants are inessential for power-law exponents and are dropped.

B.3 Case $\beta < \alpha$ (simple, latent accumulation on frequent features)

Here $\gamma > 1$, so $\sum_{i \geq 1} i^{-\gamma}$ converges to a constant $Z(\gamma)$. From (2), $N \propto \sum_{i \leq i_c} i^{-\gamma} \to Z(\gamma)$ implies the proportionality constant in (2) scales as $\kappa \propto N$. The discovery cutoff is set by $n_{i_c} \approx 1$:

$$1 \;\approx\; \kappa\, i_c^{-\gamma} \quad\Longrightarrow\quad i_c \;\propto\; \kappa^{1/\gamma} \;\propto\; N^{\frac{1}{\gamma}} = N^{\frac{1+\beta}{1+\alpha}}.$$

Thus

$$\boxed{D(N) \;\propto\; N^{\frac{1+\beta}{1+\alpha}} \quad \text{(sublinear discovery)}.} \tag{3}$$

For the loss, the discovered part scales as

$$\sum_{i \leq i_c} p_i\, n_i^{-\beta} \;\propto\; \kappa^{-\beta} \sum_{i \leq i_c} i^{-\gamma} \;\propto\; \kappa^{-\beta} \;\propto\; N^{-\beta},$$

while the undiscovered tail $\sum_{i > i_c} p_i \propto i_c^{-\alpha} \propto N^{-\alpha(1+\beta)/(1+\alpha)}$ decays faster since $\alpha > \beta$. Therefore

$$\boxed{\mathcal{L}(N) \;\propto\; N^{-\beta}.} \tag{4}$$

Intuitively, the SAE keeps shaving loss on common feature manifolds; discovery lags.

B.4 Case $\alpha < \beta$ (benign, feature discovery keeps up)

Now $\gamma < 1$, so $\sum_{i \leq i_c} i^{-\gamma} \propto i_c^{1-\gamma}$. Using $N \propto \kappa\, i_c^{1-\gamma}$ and the threshold $1 \approx \kappa\, i_c^{-\gamma}$, we eliminate κ to find

$$N \;\propto\; i_c \qquad\Longrightarrow\qquad \boxed{D(N) \;\propto\; N.}$$

For the loss over discovered features,

$$\sum_{i \leq i_c} p_i\, n_i^{-\beta} \;\propto\; \kappa^{-\beta} \sum_{i \leq i_c} i^{-\gamma} \;\propto\; i_c^{-\beta\gamma}\, i_c^{1-\gamma} \;=\; i_c^{1-(1+\alpha)} \;=\; i_c^{-\alpha} \;\propto\; N^{-\alpha}.$$

The undiscovered tail obeys $\sum_{i > i_c} p_i \propto i_c^{-\alpha} \propto N^{-\alpha}$, so both pieces match and

$$\boxed{\mathcal{L}(N) \;\propto\; N^{-\alpha}.} \tag{5}$$

Here, extra width primarily buys new features rather than over-tiling old manifolds; loss scaling mirrors the frequency tail.

C Additional Experimental Details

For Figure 3 and Figure 6, we trained SAEs on synthetic feature manifolds. For these experiments, we trained for 12000 steps, with a batch size of 2048, and a learning rate of 10^{-3} with the Adam optimizer. For the L1 penalty calculation, we use the trick of multiplying the decoder vector L2 norms by the latent activation [34].

D SAE scaling curves on synthetic manifolds

In Figure 6, we show JumpReLU SAE scaling on individual feature manifolds like we did for ReLU SAEs in Figure 3.

E SAE feature geometry on LLMs and vision models

E.1 JumpReLU Gemma Scope SAEs

If a large number of latents "tile" a single low-dimensional feature manifold, then we would expect the decoder directions for those SAE latents to have neighbors with high cosine similarity. One can see this effect directly in Figure 1, where we see that the SAE decoder latents begin to be arranged quite tightly together along the manifold. In this section, we study whether SAEs on real neural

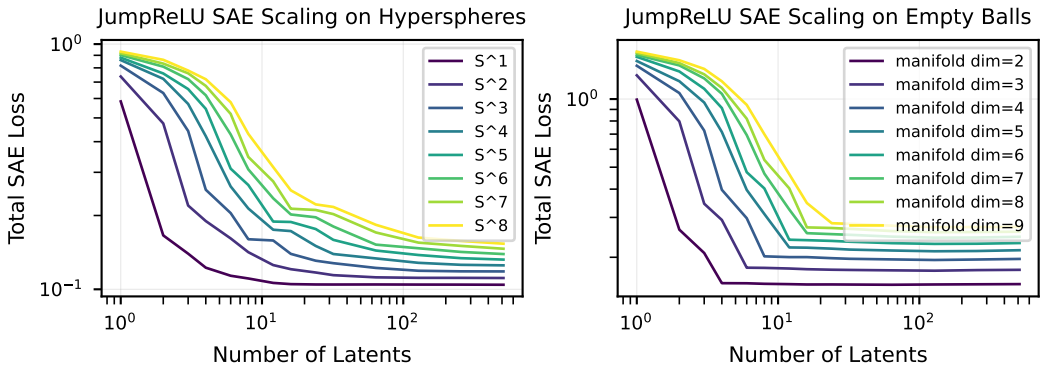

Figure 6: JumpReLU SAE scaling on individual toy feature manifolds, showing how $L(n)$ curves depend on feature geometry. We train with the tanh loss from [35] with $c = 0.1$ and $\lambda_s = 1.0$. **Left**: JumpReLU SAE scaling on unit hyperspheres of varying dimension. **Right**: JumpReLU SAE scaling on points sampled in $\{\mathbf{x} : 0.5 < |\mathbf{x}| < 2\}$. We see that, like with ReLU SAEs, that when there is variation in the radial direction between samples that our SAEs do not learn solutions which can continue to accommodate latents, and instead plateau after allocating roughly $2d_i$ latents to the feature manifold.

network activations have large numbers of latents with very high cosine similarity to their nearest neighbor.

We first study this in the Gemma Scope SAEs [16]. In Figure 7, we plot the distribution over cosine similarities between decoder latent vectors and their nearest neighbor for Gemma Scope SAEs on layer 12 (residual stream) of gemma-2-2b. While the distribution is skewed substantially higher than one would expect if all latent decoder vectors were random (and thus approximately orthogonal), we do not overall see a very large fraction of latents with extremely high cosine similarity to their nearest neighbor.

Intriguingly though, for some SAEs we do see a small uptick on the right side of this distribution, where between 10-100 latents have cosine similarity > 0.97 with their nearest neighbor. When we investigated these latents in one SAE (width_262k, average_l0_121), we found that for each of these latents, their nearest neighbor was dead (across a dataset of over 250 million tokens). These latents are not then being used to very sparsely reconstruct points on a manifold, and instead seem to be an artifact of the training process. However, the alive latents in this set are not typical SAE latents. A large number of these latents fire on tokens representing single numerical digits and single alphabet characters. We do not have an explanation of this phenomenon, but wonder whether there may be some underlying manifold representation which the SAE at one point in training tried to "tile", but then when the latents got too close, one of them was killed to reduce the L0 loss.

E.2 ReLU L1 SAEs on Inception-v1

We also study the geometry of latent decoder directions on SAEs trained on Inception-v1 activations. We train SAEs on activations from mixed3b using an L1 coefficient, λ, of 1, a learning rate of 10^{-4}, and an expansion factor of 16 (a total of 7680 latents).

In Figure 8, we find that on Inception-v1, a meaningful fraction of SAE latents have very high cosine similarity with their nearest neighbor. We note however that this could be due to latents being duplicated, which is not strongly disincentivized by the L1 loss, as pointed out in [13].

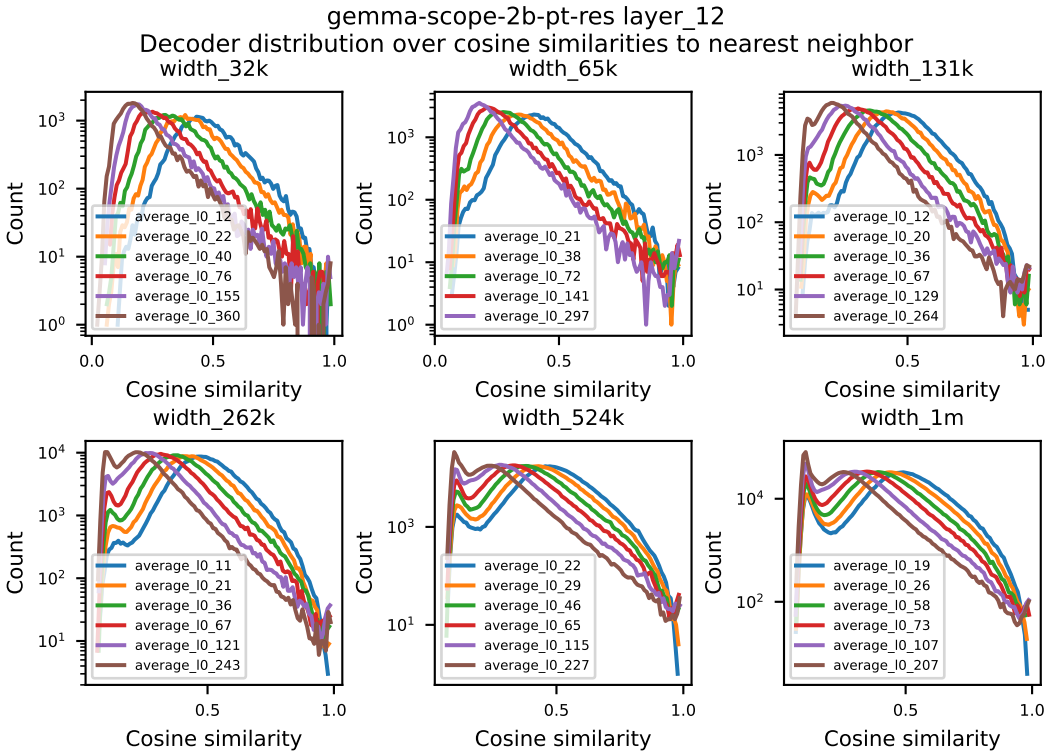

Figure 7: Distribution over cosine similarities between decoder vectors and their nearest neighbor.

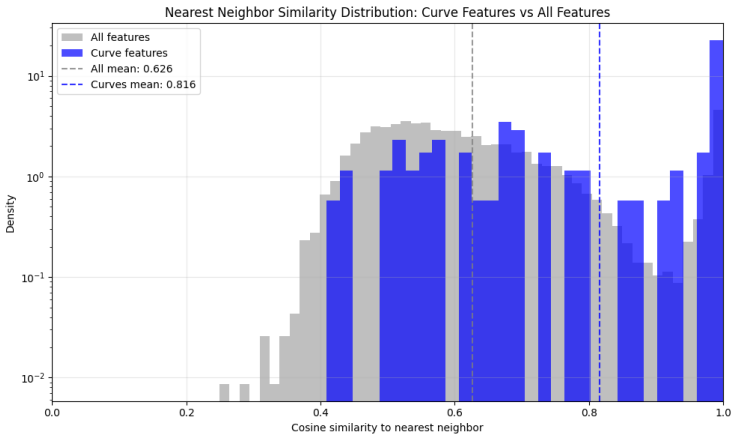

Figure 8: Distribution over pairwise cosine similarities for Inception V1.