# ToyBench: A Less Toy Model Of Superposition

**Niclas Küpper*** **Oliver Sieweke*** **Kaushik Reddy*** **Kola Ayonrinde**
LASR Labs
niclas@kupper.co

## Abstract

Benchmarks are instrumental in the development of Sparse Autoencoders (SAEs). There is a lack of benchmarks that isolate performance on individual feature structures and can reveal failure modes invisible to aggregate metrics. We introduce **ToyBench**, a synthetic benchmark consisting of eight different feature distributions, covering known and plausible phenomena in LLMs. We train autoencoders to compress these distributions into lower-dimensional representations. Autoencoder design non-trivially affects SAE performance, such as F1 scores, even for simple feature structures. No SAE dominates across the whole benchmark, even standard ReLU SAEs perform best on some distributions. ToyBench complements existing benchmarks by providing per-structure diagnostics that pinpoint SAE failure modes.

## 1 Introduction

Understanding the internal representations of Large Language Models (LLMs) has been the central goal of mechanistic interpretability. Models represent more features than they have embedding dimensions in a phenomenon known as superposition Elhage et al. [2022b]. This obfuscates the interpretability of the embedding space.

Sparse Autoencoders (SAEs) have emerged as a promising tool for extracting features from superposition Bricken et al. [2023], Gao et al. [2024]. However, they still exhibit a variety of failure modes, such as splitting, absorption, hedging, incomplete recovery, and misattribution, even in simplified settings Bricken et al. [2023], Chanin et al. [2025b,a], Chanin and Garriga-Alonso [2026].

In LLMs the lack of ground-truth features and limited knowledge of their geometric structure, beyond isolated examples, makes it difficult to precisely evaluate and improve SAE architectures. We use autoencoders as toy models.[2] This provides a controlled setting, where ground-truth features are known by construction and in which feature distributions can be varied and studied systematically Elhage et al. [2022b], Henighan et al. [2023], Jermyn et al. [2022]. Moreover, known ground truth allows us to define metrics that directly capture interpretability properties we care about, rather than relying on proxies, such as autointerpretability scores Paulo et al. [2025] and concept disentanglement Karvonen et al. [2024].

SynthSAEBench Chanin and Garriga-Alonso [2026] has made an important contribution by providing a benchmark of synthetic ground-truth features with correlational and hierarchical structures. We believe this work can be complemented by focusing on smaller stylized feature distributions to perform fine-grained assessments and better disaggregate failure modes.

We propose *ToyBench*, a benchmark containing eight different feature structures capturing known and plausible phenomena in LLMs, including multi-dimensional features, and features living on

*Equal contribution.

[2]We point out that we differentiate between sparse autoencoders (SAEs) and autoencoders (AEs). When referring to autoencoders (or AEs) we will always mean the toy model of the residual stream.

Preprint. Under review.

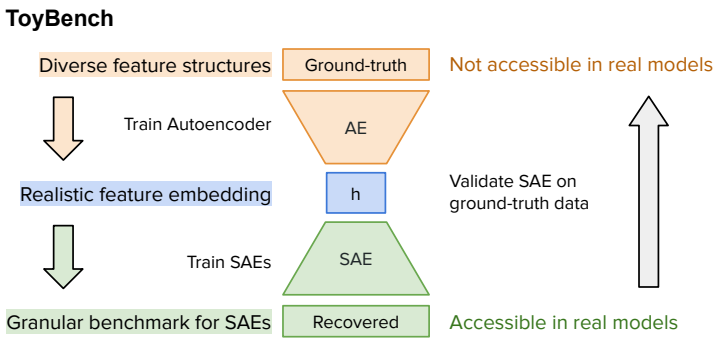



Figure 1: ToyBench provides a suite of stylized feature distributions to test SAEs. The synthetic setup gives access to ground-truth that is inaccessible in real models. The autoencoder (AE) enforces the linear representation hypothesis and is trained to be more realistic.

manifolds. To obtain realistic embedding spaces, we train toy autoencoders to compress these feature distributions into lower dimensional spaces.

**Contributions**

**Distributions** There exists a sharp threshold for parent-child cosine-similarity in hierarchical distributions. We also show that simplices are a possible explanation for feature splitting. Finally, we observe that across all distributions the feature correlation and interference have a positive relationship (Section 4).

**Impact of training the AE** We show that the relative positioning of features in the embedding space affects SAE reconstruction. SAE F1 scores are $\sim 50\%$ higher on feature arrangements trained by autoencoders than on randomly constructed feature directions (Section 5.1). We argue that trained distributions should be preferred as they more plausibly mirror what arises in real networks and carry structure that SAEs can leverage.

**Benchmark** We assemble *ToyBench*, a selection of trained distributions covering different aspects of known or plausible feature structures into a benchmark. We report and compare performance metrics for current state-of-the-art SAEs. No SAE architecture reaches the best theoretical performance. Furthermore, no SAE dominates across all distributions (Section 5.2).

**Library** We release `occhio`[3], the Python library used to generate our benchmark. It enables researchers to train autoencoders on their own variations of feature distributions, explore the resulting embedding spaces, and evaluate interpretability tools on them (Section 5.3).

## 2 Background

### 2.1 Features

We use the word *feature* to refer to an identifiable high-level concept. In neural networks, features are concept whose representation helps reduce the training loss and, in many settings, improve generalization. In our toy model setting, we can precisely control the features by construction.

When training a neural network, it may need to represent more features $f_1, f_2, \ldots, f_N$, than it has dimensions $d$. In this case it will not be able to allocate a dedicated dimension to each feature. Instead, the model typically learns to represent more features than it has dimensions by placing the features non-orthogonally in the intermediate layer. This is called *superposition*.

[3]Available at https://docs.occhio.dev.

Superposition leads to *interference* between features, i.e. reading from one feature might also activate other unrelated features. Non-linear activations can suppress this interference, and together with feature sparsity, are required for superposition to emerge Elhage et al. [2022b]. These phenomena are well established and have been observed in the residual stream of LLMs Scherlis et al. [2022], Elhage et al. [2022a], Henighan et al. [2023], Elhage et al. [2022b].

### 2.2 Toy models

Initial work studying feature superposition relied on simplified assumptions about the feature distribution, primarily studying superposition using features that are independently and identically distributed.

More specifically, Elhage et al. [2022b], Henighan et al. [2023] study input features $f_i \sim \text{SparseUniform}(p, [0,1]) := \text{Bernoulli}(p) \times \text{Uniform}([0,1])$, which we refer to as the *sparse uniform* distribution. We will usually consider the situation where $p_i \sim (1+i)^{-\alpha}$. We call this the *zipfian* distribution.

They encode this distribution using a simple autoencoder with tied encoder and decoder weights $W \in \mathbb{R}^{d \times N}$, where $N \in \mathbb{N}$ is the feature dimension, and $d < N$ is the dimension of the embedding space. Tying weights forces the decoder to read from where the encoder has written, makes directions in embedding space unambiguous, and empirically improves autoencoder performance.

$$\begin{aligned} \text{Encoder: } h = Wf \qquad \text{Decoder: } \widehat{x} &= \text{ReLU}(W^\top h + b) \\ &= \text{ReLU}(W^\top W f + b). \end{aligned} \tag{1}$$

We use the above autoencoder setup throughout this paper.

Previous work has shown that despite their simplicity the above model can exhibit surprising complexity. For any given feature the model can choose not to represent it, represent it orthogonally to all others, or place it in superposition with other feature(s). This phase change is sharp Elhage et al. [2022b].

Instead of training autoencoders, some previous work has used randomly initialized weight matrices $W$ Chanin and Garriga-Alonso [2026], Chen et al. [2026]. To improve on this, Chanin and Garriga-Alonso [2026] perform several orthogonalization steps after initialization, independent of any training data. We refer to such autoencoders as *constructed*.

### 2.3 Sparse Autoencoders

Sparse Autoencoders (SAEs) have emerged as a tool to disentangle superposition by decomposing the original representation into a *sparse* set of latent feature directions. These sparse firings are more interpretable than the underlying space. In this sense the SAE can be seen as the inverse to the autoencoder.

Let $d_{\text{dict}} \in \mathbb{N}$ be the dimension of the SAE latent space. Let $V \in \mathbb{R}^{d_{\text{dict}} \times d}$ and $D \in \mathbb{R}^{d \times d_{\text{dict}}}$ be the SAE encoder and decoder. $D$ is also called the *dictionary* or *decoder* matrix. The sparse latent $s \in \mathbb{R}^{d_{\text{dict}}}$ and the reconstruction $\hat{x}$ are computed as $s = \text{ReLU}(V(x - b_d) + b_e)$ and $\hat{x} = Ds + b_d$. When training the SAE we use the loss function $\|x - \hat{x}\|_2^2 + \lambda \|s\|_1$, where we enforce the columns of $D$ to have unit norms. This prevents the SAE from trivially decreasing the $L^1$ penalty by shrinking $\|s\|_1$ through rescaling.

**Problems** Despite assuming the Linear Representation Hypothesis (LRH), SAEs do not work well even in situations where the target space follows LRH by construction Chanin and Garriga-Alonso [2026]. When the feature distribution departs from i.i.d. assumptions, such as those with correlational, conditional, hierarchical, compositional, or relational structure, SAE performance degrades further still.

For example, Bricken et al. [2023] show that SAEs trained on hierarchical distributions are susceptible to feature splitting. Related, Chanin et al. [2025b] demonstrate feature absorption, and Anders et al. [2024] show feature composition. As a result, the success of sparse dictionary methods is influenced by the statistical and compositional structure of the features that gave rise to the original representation.

The study of these failure modes in toy models, and the geometric structures that drive them, remains underexplored. We study a family of parametrizable feature distributions and investigate how the geometry of learned representations affects downstream SAE performance, extending the results of Elhage et al. [2022b] and Chanin et al. [2025b] to correlated, hierarchical, and higher-dimensional settings.

## 3 Methods

We use two families of metrics: embedding space characterizations that describe the learned feature geometry independently of any dictionary, and SAE metrics that measure how well a trained SAE captures the underlying features. For SAE evaluation, following O'Neill et al. [2025], Chanin and Garriga-Alonso [2026], we match ground-truth features to dictionary elements using an optimal bipartite matching $M$ that maximizes MCC.[4] For the $F_1$ score, we count a true positive (TP) when an SAE latent fires together with its matched feature; false positives (FP) and false negatives (FN) are defined analogously.

Table 1 summarizes the metrics we report. All MCC and $F_1$ depend on the matching $M$.

| Metric | Definition | Notes |
|---|---|---|
| Embedding norm | $\|W_i\|$ | Norm of feature $i$'s embedding vector. Usually features cluster near 1 (learned) or 0 (abandoned). |
| Interference | $I_{ij} = (\hat{W}_i \cdot W_j)^2$ | Overlap between features $i$ and $j$. Total interference $\sum_{j \neq i} I_{ij}$ captures how much feature $i$ interacts with all others. |
| MCC | $\frac{1}{\min(N, d_{\text{dict}})} \sum_{(i,j) \in M} \frac{W_i \cdot D_j}{\|W_i\| \|D_j\|}$ | Measures the alignment of the dictionary directions with the AE directions. |
| $R^2$ | $1 - \frac{\mathbb{E}[\|x - \hat{x}\|_2^2]}{\text{Var}(x)}$ | How well $\hat{x}$ reconstructs the embedding space. $R^2 = 1$ corresponds to perfect reconstruction. |
| $F_1$ | $\frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$ | Measures predictive performance. Harmonic mean of precision and recall. Computed per-latent and then averaged. |

Table 1: Metrics used throughout the paper. The top block characterizes the embedding space; the bottom block evaluates SAEs, where $M$ is the optimal bipartite matching between features and dictionary elements.

## 4 Distributions

Table 2 summarizes the synthetic feature distributions used in this paper and the structural phenomenon each one targets. Full definitions and justifications are given in Appendix A.

For the purpose of building intuition we will investigate the Hierarchical Pairs and Simplicial Complex distributions in additional detail. When referring to the embedding geometry we are referring to the embedding learned by a trained autoencoder (Equation 1).

### 4.1 Hierarchical Pairs

Geometrically, we notice that parent-child pairs prefer to be roughly orthogonal, and the feature directions collapse quickly if there is not enough space for them to be orthogonal. The intuition is that samples in the latent space are $45°$ apart when the feature vectors are orthogonal (Figure 2a). If the feature vectors are closer together, the autoencoder cannot differentiate them and collapses the features.

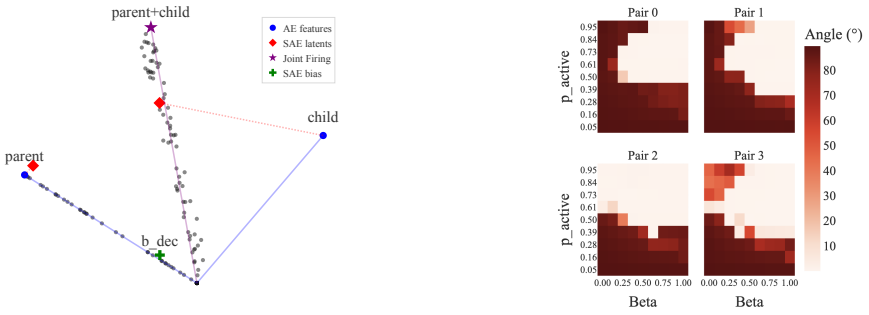
[4] See Appendix B for more details on matching features.

| Distribution | Purpose |
|---|---|
| Zipfian | Serves as the baseline distribution: sparse uniform features with power-law firing probabilities $p_i \propto (1+i)^{-\alpha}$. |
| Correlated Pairs | Isolate the effect of pairwise correlations, which have been observed between features in LLMs. |
| Hierarchical Pairs | Models parent-child feature hierarchies seen in LLMs. |
| Preferential Attachment | Power-law feature connectivity and cascaded activations. Mimics the scale-free interaction networks. |
| Simplicial Complex | The natural geometry for probability distributions and for triangulated approximations of smooth manifolds. |
| Deep Hierarchy | Generalize hierarchical pairs to deeper DAGs in which features may have multiple parents and activations propagate upward to the root. |
| Spherical | Place features on a sphere to study how SAEs handle low-dimensional continuous feature manifolds. |
| Torus | Extend the spherical setup to a torus, probing SAE behavior on product-manifold geometries with more complex topology. |

Table 2: Synthetic feature distributions used in this paper, together with the phenomenon each one is designed to probe.

The same geometry also provides insight into the phenomenon of absorption Chanin et al. [2025b]. Absorption occurs exactly when the SAE learns the joint parent + child direction instead of the child direction. We also validate their finding that full absorption only occurs when the spread in the parent + child direction is sufficiently low. The spread is controlled by $\beta$, where $\beta = 0$ corresponds to no correlation in the firing magnitude, i.e. the widest spread.

Looking at the parent-child separation when multiple hierarchical pairs must be learned, we find a sharp phase transition in firing probability and $\beta$, which controls the similarity of firing magnitudes (Figure 2b). Hierarchical pairs take up more space compared to the sparse uniform distribution, as their feature directions require more separation.




(a) The SAE's child latent absorbs the parent direction rather than learning the child alone. Samples lie along parent-only and parent+child axes, which higher $\beta$ concentrates.

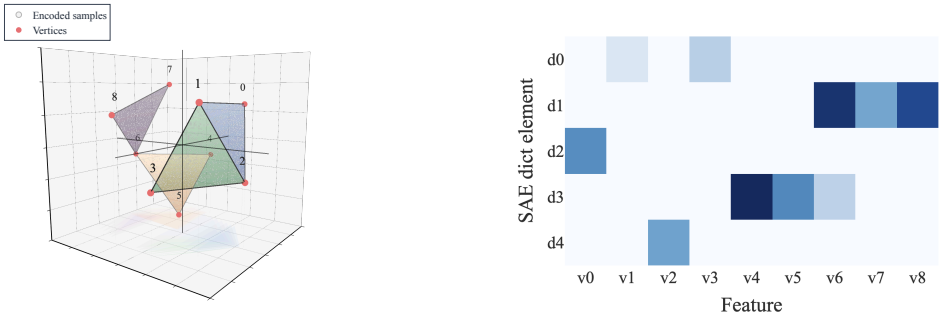
(b) Parent-child angle exhibits a sharp phase transition in $(\beta, p)$: pairs flip from orthogonal to collapsed. 8 features in 4 dimensions.

Figure 2: Hierarchical Pairs: the SAE absorbs parents into child latents (left), and parent-child separation undergoes a sharp phase transition (right).

### 4.2 Simplicial Complex Distribution

We find that the autoencoder learns to represent simplices as equilateral triangles when possible (Figure 3a). When simplices are glued together, each simplex tends to be oriented so that the line from the origin to its centroid is normal to the simplex surface, i.e. each simplex is approximately tangent to a sphere centered at the origin.

When an SAE does not have capacity to learn every feature, it learns to represent a whole simplex in a single dictionary entry (Figure 3b). This occurs when the SAE width is less than the total number of features.




(a) The autoencoder embeds each simplex as a near-equilateral triangle. 9 features in 3 embedding dimensions.

(b) Under-capacity SAEs collapse whole simplices into single latents. One-hot activations at the 75% quantile; simplices as in Figure 3a.

Figure 3: Simplicial Complex: the autoencoder embeds simplices as equilateral triangles (left), and under-capacity SAEs collapse whole simplices into single latents (right).

### 4.3 General Observations

Across all distributions studied in this paper (except Zipfian), we observe a positive relationship between the empirical correlation of features and the interference of their learned representation in the trained autoencoder. Table 4 shows this relationship for each distribution.

This is intuitive: correlated features are likelier to fire together, so the model incurs less loss from placing them in higher interference configurations. The sparse uniform distribution is the exception, as it has no correlations between features by construction, and accordingly the measured trend is zero.

| Distribution | Zipfian | Corr. Pairs | Hier. Pairs | Deep Hier. | Pref. Att. | Simplicial | Spherical | Torus |
|---|---|---|---|---|---|---|---|---|
| Corr % | 0.0 | 2.9 | 3.7 | 19.0 | 10.3 | 22.3 | 11.8 | 13.5 |
| 95% CI | ±0.5 | ±0.5 | ±0.4 | ±0.4 | ±0.5 | ±0.4 | ±0.4 | ±0.4 |

Figure 4: Empirical feature correlation is positively associated with learned-embedding interference for every distribution except Sparse Uniform, which has no correlations by construction. Values are correlations with 95% CI.

## 5 Results

In Section 5.1 we compare constructed with trained autoencoders on the Zipfian distribution. Our findings will motivate some of the design decisions of the benchmark, which we discuss in Section 5.2.

### 5.1 Constructed and Trained Autoencoders

SynthSAEBench Chanin and Garriga-Alonso [2026] made meaningful progress in evaluating SAEs against toy models with known ground-truth feature structure, enabling principled measurement of failure modes that are difficult to quantify in natural language data.

One way we depart from their approach is by training the autoencoder in Equation 1. SynthSAEBench constructs $W$ by orthonormalizing after random initialization.[5] This distinction, as we show below,

[5]Additionally, we allow all autoencoders, to learn decoder bias $b$. This does not affect the embedding space directly, but it does mediate its behavior. See Appendix C.

meaningfully affects the downstream performance of an SAE trained to reconstruct the toy model embedding space, *even in the simplest zipfian case*. We explore the geometry that causes this behavior in Appendix C.

**SAE comparison** To test SAE performance on trained vs constructed autoencoders we run a sweep across target $L^0_{SAE}$[6] (Figure 5). We also include a trained autoencoder with unit-norm columns in $W$ as an additional ablation, henceforth 'unit-norm'. For any fixed $L^0_{SAE}$ the SAEs fitted to trained autoencoders have a higher $F_1$ and $MCC$ score. At the peak the SAEs fitted to trained autoencoders outperform by around 50%.

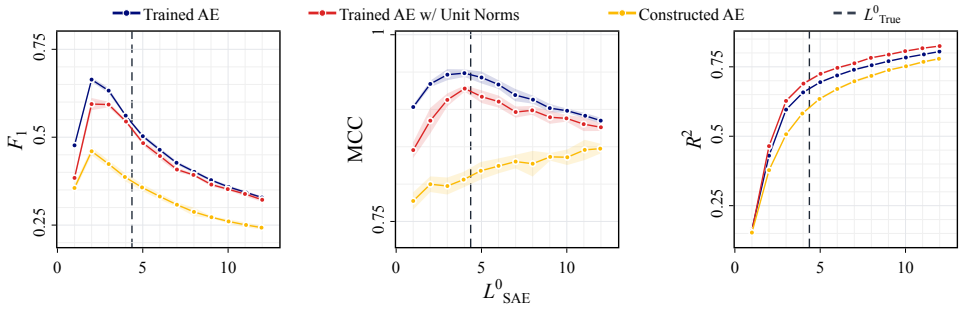



Figure 5: ReLU SAEs perform better on trained AEs relative to constructed AEs across all metrics. Run across 5 seeds. Shaded area is stdev. $N = 500$, $d = 64$ and SAE-width is 250.

Most of the performance gain is explained by the angular structure, since the SAEs fitted to unit-norm autoencoders perform similar to the trained. The unit-norm autoencoder results in higher $R^2$, but is worse at $F_1$ score and MCC, further demonstrating that better $R^2$ does not necessarily lead to better probing. Peak $F_1$ performance is achieved when $L^0_{\text{SAE}} \leq L^0_{\text{True}}$ for all autoencoders. This trend continues to hold for the SAEs we test in Section 5.2.

The key takeaway is that the geometry beyond 'roughly-orthogonal' has real downstream effects on SAE performance.

### 5.2 Benchmark

Each benchmark distribution is produced by training the representation of 1296-dimensional feature structures into a 200-dimensional embedding space. The sizes were chosen such as to allow the autoencoder to represent all features. The distributions were tuned to an $\mathbb{E}[L^0] \approx 5$, consistent with the linear accessibility bound Garg et al. [2026].

For each distribution, we train ReLU, Matching Pursuit, BatchTopK, and Matryoshka BatchTopK SAEs (adopting the naming from SAEBench Karvonen et al. [2025]). Each of these SAEs has width 648 and is trained with batch-size 1024 for $XXX$ samples. In Table 3 we report the results at $L^0 = 3$, which generally yielded the best performances (Appendix F reports versions of the table directly selecting the best $F_1$ and MCC.).

Our setting allows us to use the autoencoder unembedding (Equation 1) as a baseline, by converting the ReLu to a JumpReLU activation. To do this we sweep across activation thresholds to maximize F1 scores. No SAE achieves an optimal $F1$ score across all distributions (see Figure 6). Without access to ground truth one might be tempted to explain away performance limitations in LLMs with the purported complexity of the representation space. The results show that SAEs can struggle even on stylized distributions.

Further, the architectures display strengths and weaknesses in different domains. While the standard $ReLU$ SAE underperforms significantly on correlated and hierarchical distributions, it holds up well simplicial complex distributions and tends to peak closer to the true $L0$ value of 5.

Notably, Matryoshka SAEs consistently slightly underperforms BatchTopK SAEs, even on hierarchical pairs despite being designed for nested structures in mind.

[6] $L^0$ targeting is explained in Appendix E

| | ReLU | | | Matryoshka | | | BatchTopK | | | MatchingPursuit | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dist. | $F_1 \uparrow$ | MCC $\uparrow$ | $R^2 \uparrow$ | $F_1 \uparrow$ | MCC $\uparrow$ | $R^2 \uparrow$ | $F_1 \uparrow$ | MCC $\uparrow$ | $R^2 \uparrow$ | $F_1 \uparrow$ | MCC $\uparrow$ | $R^2 \uparrow$ |
| Zipf | 0.58 | 0.73 | 0.45 | **0.74** | **1.00** | **0.78** | **0.74** | **1.00** | **0.78** | 0.68 | 0.94 | 0.74 |
| CP | 0.46 | 0.62 | 0.38 | 0.72 | 0.83 | 0.80 | **0.74** | **0.86** | **0.82** | 0.67 | 0.82 | 0.78 |
| HP | 0.48 | 0.63 | 0.46 | 0.71 | 0.87 | 0.88 | **0.74** | **0.88** | **0.89** | **0.74** | 0.85 | 0.85 |
| DH | 0.51 | 0.69 | 0.50 | 0.65 | 0.97 | **0.75** | 0.65 | **0.98** | **0.75** | **0.75** | 0.92 | 0.72 |
| PA | **0.66** | **0.96** | 0.50 | 0.65 | 0.95 | 0.61 | 0.65 | 0.95 | **0.62** | 0.65 | 0.94 | 0.61 |
| SC | **0.73** | **0.89** | 0.69 | 0.72 | 0.88 | 0.78 | **0.73** | **0.89** | **0.80** | 0.64 | 0.88 | 0.78 |
| $S^n$ | 0.65 | 0.81 | 0.79 | 0.65 | 0.81 | 0.83 | 0.69 | **0.83** | **0.86** | **0.73** | **0.83** | **0.86** |
| Torus | 0.47 | 0.71 | 0.55 | 0.65 | 0.83 | 0.73 | 0.70 | 0.86 | **0.76** | **0.74** | **0.88** | **0.76** |
| Mean | 0.57 | 0.75 | 0.54 | 0.69 | 0.89 | 0.77 | **0.71** | **0.91** | **0.78** | 0.70 | 0.88 | 0.76 |

Table 3: Performance of sparse autoencoders at a fixed sparsity of $L_0 = 3$. Distributions: Zipf = Zipfian, CP = Correlated Pairs, HP = Hierarchical Pairs, DH = Deep Hierarchy, PA = Preferential Attachment, SC = Simplicial Complex, $S^n$ = Spherical. Best result across SAE settings per distribution is shown in **bold**. Colors highlight the best and worst scores on a metric across all distributions.

We emphasize that relative performance across distributions is more informative than aggregated scores. The mean can reveal general trends but should not be used as a definitive ranking.

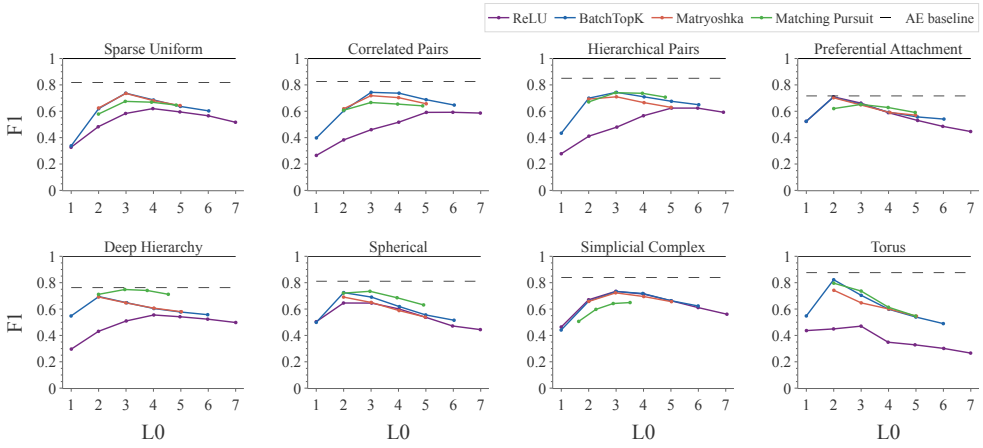



Figure 6: Notice that the autoencoder baseline is not reached for most distributions. $F_1$ scores for each distribution and each SAE architecture across multiple $L^0$.

### 5.3 Library

The benchmark was produced with `occhio`, a lightweight package built on `torch` for studying superposition in toy models. While we had to make choices as to which distributions to include, the library provides a wider collection of parametrizable feature distributions and autoencoders. It can also be used to train custom distributions. This allows the user to explore the space, test further hypotheses, and create their own challenges. We provide example scripts for replicating the original TMoS paper, feature absorption, as well as the experiments from this paper.

## 6 Discussion

### 6.1 Limitations

**Feature structure in natural data** While we consider the toy distributions a useful starting point for understanding the geometry of superposition, they are simplified and do not fully capture the complexity of natural data. Future work could empirically analyze feature structure in natural language data and then construct new distributions that more closely match observed properties (such

as correlations, hierarchy, or manifold structure) from real datasets. This would make it possible to evaluate whether failure modes and behaviors observed in toy models also appear in practice, concretely test the relevance of specific synthetic benchmarks, and iteratively refine interpretability tools towards practical impact.

There is emerging evidence for the presence of such structures and failure modes in natural settings. For example, Bussmann et al. [2025] provides strong indications of hierarchical feature structures in real data. Moreover, Chanin et al. [2025b] and Chanin et al. [2025a] show feature absorption and hedging, which are further evidence of hierarchy and correlation. Previous work by Engels et al. [2025] shows that language model features may not be one-dimensionally linear.

**Understanding superposition as computation** Whereas our toy models show superposition purely as compression, superposition in the transformer residual stream imposes additional computational demands. This means the superposition structures observed in toy models do not fully reflect how real networks organize their latent spaces. More complex feature structures may partially address this by imposing richer geometric constraints, but they still lack computational pressure. A complementary approach is to study how computational targets shape latent representations directly, as in Nanda et al. [2023], Chen et al. [2026].

**Metrics** The metrics available to us for measuring SAE performance do not capture everything we might care about. Both $F_1$ and MCC rely to some extent on the notion of 1D features, which multi-dimensional and manifold features do not necessarily adhere to.

### 6.2 Related Work

**SynthSAEBench** The closest existing work is SynthSAEBench Chanin and Garriga-Alonso [2026]. Fundamentally, our approaches are similar, but our focus is different. On the one hand, ToyBench aims to be more realistic by training the encoder. On the other, it provides higher fidelity feedback through a variety of distributions.

**Other work inspired by Toy Models of Superposition** Following Elhage et al. [2022b], subsequent work was optimistic about finding fully monosemantic decompositions of models or training models to tend towards monosemanticity Jermyn et al. [2022]. There is ongoing work to use toy models as a test-bed for interpretability methods, such as sparse transformers Gao et al. [2025].

**Extensions and other work** Other notable toy models include the work by Nanda et al. [2023] on a transformer learning modular addition and Michaud's work on quanta Michaud et al. [2024], Michaud [2026]. Shai et al. showed that transformers are able to represent their belief about hidden states of hidden Markov models in their residual stream Shai et al. [2025]. Their work was performed on networks without superposition, and we echo their call to investigate belief state geometry in compressed domains.

### 6.3 Future Work

Toy models are a powerful tool for understanding the geometry of superposition because they provide a controlled setting with access to ground truth. This makes it possible both to study representational geometry directly and to benchmark interpretability tools against *known* conditions. We hope this paper encourages further exploration of superposition across a broader range of controlled settings.

The present work could be naturally extended to different assumptions about data distributions, such as more diverse variants of multi-dimensional input features. Beyond data distributions, we hope that `occhio` will facilitate more systematic and principled evaluation of interpretability methods. More ambitiously, access to a clean sandbox with known ground truth could be used to improve interpretability methods with legible success and failure conditions.

If toy models turn out to be a cheap approximation of larger SAE benchmarks such as SAEBench Karvonen et al. [2025], such techniques could speed up SAE development. Either way, toy models might help to understand which exact behaviors are (or are not) captured by SAEs Hindupur et al. [2025].

## Acknowledgments and Disclosure of Funding

We thank David Chanin and Eric Michaud for their insightful conversations.

Thanks to LASR and Arcadia Impact for providing funding. Special thanks to Erin Robertson, Brandon Riggs and Tamkeen Nawab.

## References


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, 2024.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286 (5439):509–512, 1999. doi:10.1126/science.286.5439.509. URL https://www.science.org/doi/abs/10.1126/science.286.5439.509.

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, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning Multi-Level Features with Matryoshka Sparse Autoencoders, March 2025. URL http://arxiv.org/abs/2503.17547. arXiv:2503.17547 [cs].

David Chanin and Adrià Garriga-Alonso. SynthSAEBench: Evaluating Sparse Autoencoders on Scalable Realistic Synthetic Data, 2026. URL https://arxiv.org/abs/2602.14687.

David Chanin, Tomáš Dulka, and Adrià Garriga-Alonso. Feature hedging: Correlated features break narrow sparse autoencoders, 2025a. URL https://arxiv.org/abs/2505.11756.

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, November 2025b. URL http://arxiv.org/abs/2409.14507. arXiv:2409.14507 [cs].

Zixin Jessie Chen, Hao Chen, Yizhou Liu, and Jeff Gore. Superposition unifies power-law training dynamics, 2026. URL https://arxiv.org/abs/2602.01045.

Nelson Elhage, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer ElShowk, Nicholas Joseph, Nova DasSarma, Ben Mann, Danny Hernandez, Amanda Askell, Kamal Ndousse, Andy Jones, Dawn Drain, Anna Chen, Yuntao Bai, Deep Ganguli, Liane Lovitt, Zac Hatfield-Dodds, Jackson Kernion, Tom Conerly, Shauna Kravec, Stanislav Fort, Saurav Kadavath, Josh Jacobson, Eli Tran-Johnson, Jared Kaplan, Jack Clark, Tom Brown, Sam McCandlish, Dario Amodei, and Christopher Olah. Softmax linear units. *Transformer Circuits Thread*, 2022a. https://transformer-circuits.pub/2022/solu/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, September 2022b. URL http://arxiv.org/abs/2209.10652. arXiv:2209.10652 [cs].

Joshua Engels, Eric J. Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear, 2025. URL https://arxiv.org/abs/2405.14860.

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, June 2024. URL http://arxiv.org/abs/2406.04093. arXiv:2406.04093 [cs], version: 1.

Leo Gao, Achyuta Rajaram, Jacob Coxon, Soham V. Govande, Bowen Baker, and Dan Mossing. Weight-sparse transformers have interpretable circuits, November 2025. URL http://arxiv.org/abs/2511.13653. arXiv:2511.13653 [cs].

Nikhil Garg, Jon Kleinberg, and Kenny Peng. How many features can a language model store under the linear representation hypothesis?, 2026. URL https://arxiv.org/abs/2602.11246.

Allen Hatcher. *Algebraic topology*. Cambridge University Press, Cambridge, 2002. ISBN 0-521-79160-X; 0-521-79540-0.

Tom Henighan, Shan Carter, Tristan Hume, Nelson Elhage, Robert Lasenby, Stanislav Fort, Nicholas Schiefer, and Christopher Olah. Superposition, memorization, and double descent. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/toy-double-descent/index.html.

Sai Sumedh R. Hindupur, Ekdeep Singh Lubana, Thomas Fel, and Demba Ba. Projecting Assumptions: The Duality Between Sparse Autoencoders and Concept Geometry, March 2025. URL http://arxiv.org/abs/2503.01822. arXiv:2503.01822 [cs] version: 1.

Adam S. Jermyn, Nicholas Schiefer, and Evan Hubinger. Engineering Monosemanticity in Toy Models, November 2022. URL http://arxiv.org/abs/2211.09169. arXiv:2211.09169 [cs].

Adam Karvonen, Can Rager, Samuel Marks, and Neel Nanda. Evaluating sparse autoencoders on targeted concept erasure tasks, 2024. URL https://arxiv.org/abs/2411.18895.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum McDougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. SAEBench: A Comprehensive Benchmark for Sparse Autoencoders in Language Model Interpretability, June 2025. URL http://arxiv.org/abs/2503.09532. arXiv:2503.09532 [cs].

Eric J. Michaud. On neural scaling and the quanta hypothesis. https://ericjmichaud.com/quanta/, 2026. Blog post.

Eric J. Michaud, Ziming Liu, Uzay Girit, and Max Tegmark. The Quantization Model of Neural Scaling, January 2024. URL http://arxiv.org/abs/2303.13506. arXiv:2303.13506 [cs].

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability, October 2023. URL http://arxiv.org/abs/2301.05217. arXiv:2301.05217 [cs].

Chris Olah and Josh Batson. Feature manifold toy model. Transformer Circuits Thread, May 2023. URL https://transformer-circuits.pub/2023/may-update/index.html#feature-manifolds. May 2023 Update.

Charles O'Neill, Alim Gumran, and David Klindt. Compute optimal inference and provable amortisation gap in sparse autoencoders, 2025. URL https://arxiv.org/abs/2411.13117.

Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models, 2025. URL https://arxiv.org/abs/2410.13928.

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.

Adam S. Shai, Sarah E. Marzen, Lucas Teixeira, Alexander Gietelink Oldenziel, and Paul M. Riechers. Transformers represent belief state geometry in their residual stream, February 2025. URL http://arxiv.org/abs/2405.15943. arXiv:2405.15943 [cs].

Xiangchen Song, Aashiq Muhamed, Yujia Zheng, Lingjing Kong, Zeyu Tang, Mona T. Diab, Virginia Smith, and Kun Zhang. Position: Mechanistic interpretability should prioritize feature consistency in saes, 2025. URL https://arxiv.org/abs/2505.20254.

Federico Tiblias, Irina Bigoulaeva, Jingcheng Niu, Simone Balloccu, and Iryna Gurevych. Hypothesis-driven feature manifold analysis in llms via supervised multi-dimensional scaling, 2026. URL https://arxiv.org/abs/2510.01025.

# A Distributions

We rigorously define every distribution in Section 3. The parameters for each distribution will be at the bottom of its section. Let $N = 1296$ be the number of features and $d = 200$ be the hidden dimension. We target a $E[L^0_{\text{True}}] = 8$ for each distribution.

**Zipfian** Inspired by Toy Models of Superposition Elhage et al. [2022b], Henighan et al. [2023].

Our input features fire according to $f_i \sim \text{SparseUniform}(p, [0, 1]) := \text{Bernoulli}(p) \times \text{Uniform}([0, 1])$, which we refer to as the *sparse uniform* distribution.

For the benchmark we set $h = 0.5$, $l = 2.0/N$ $p_i = h \cdot (1 + i)^{-\alpha}$, with $\alpha = \frac{\log(h)-\log(l)}{\log(N)}$

**Correlated Pairs** Correlations between features have been observed in LLMs Chanin et al. [2025a]. They also demonstrate the effect of correlation on SAEs in synthetic and real setups.

Let $N \in \mathbb{N}$ be an even number. This distribution is arranged in $N/2$ independent pairs. We define the activation probability $p_a \in (0, 1]$ and the individual firing probability $p_i \in (0, 1]$.

Each pair fires with probability $p_a$. Conditional on a pair firing, each individual feature in the pair fires with probability $p_i$. The effective density $D$ is then $p_a p_i$. The correlation between the individuals in a pair is $c = \frac{p_i(1-p_a)}{1-p_a p_i}$. Note that for any two of $p_a$, $p_i$, $D$, $c$, the remaining two can be derived.

For the benchmark we chose $p_a = h/(i + 1)^\alpha$, and $p_i \sim \text{Uniform}(0.5, 1.0)$, where $h = 0.6$, $l = 2.55/N$ and $\alpha = \frac{\log(h)-\log(l)}{\log(N)}$.

**Hierarchical Pairs** Hierarchical features occur in nature, and have in particular been observed in LLM features Chanin et al. [2025b].

Features are arranged in parent–child pairs $(2i, 2i+1)$. The parent fires independently with probability $p_a$; conditional on the parent firing, the child fires with probability $p_f$, and is otherwise zero. Active parents take $\text{Uniform}(0, 1)$ magnitudes. The child's magnitude depends on a coupling constant $\beta \in [0, 1]$: $v_{\text{child}} = \beta v_{\text{parent}} + (1 - \beta)U$, where $U \sim \text{Uniform}(0, 1)$. Pairs are mutually independent, but within each pair the child is causally downstream of the parent.

The activation correlation within a pair is $c = \sqrt{p_f(1 - p_a)/(1 - p_f p_a)}$, approaching $\sqrt{p_f}$ as $p_a \to 0$; inverting gives $p_f = c^2/(1 + p_a(c^2 - 1))$ for a target $c$.

For the benchmark we chose $p_a = h/(i + 1)^\alpha$, and $\beta \sim \text{Uniform}(0.0, 1.0)$, where $h = 0.6$, $l = 2.39/N$, $\alpha = \frac{\log(h)-\log(l)}{\log(N)}$ and $p_f = 0.6$.

**Preferential Attachment** This approach was inspired by Barabási and Albert [1999], which finds that many real world connections follow power-law degree distributions. We construct a directed graph based on this insight.

Features are arranged as nodes in a directed graph with power-law in-degree distribution. The per-edge connection probability for edges targeting node $i$ is

$$p_i = \min\left(1, p_{\text{edge}} \cdot \left(\frac{N - i}{N}\right)^\alpha\right), \tag{2}$$

so node 0 receives the most in-edges (in expectation) and node $N-1$ the fewest. We can recover the Erdős-Rényi graph by setting $\alpha = 0$.

Each node fires independently with probability $p_a$. Nodes propagate activation one step along their out-edges independently with probability $p_c$. When a node has $k$ independently-fired parents, its probability of being cascade-triggered is $1 - (1 - p_c)^k$. Active nodes take values drawn from $\text{Uniform}(0, 1)$.

For the benchmark we chose $p_a = 4/N$, $\alpha = 0.75$, $p_{\text{edge}} = 4.5/N$, and $p_c \sim \text{Uniform}(0.2, 0.6)$.

**Simplicial Complex** For our purposes, simplices are the simplest multi-dimensional feature. Intuitively, the $k$-simplex $\triangle_k$ is a $k$-dimensional triangle spanned by $k + 1$ points.

Formally, a simplex is defined as

$$\triangle_k := \left\{ a \in \mathbb{R}^{k+1}_{\geq 0} \mid \sum_{i=0}^{k} a_i = 1 \right\}. \tag{3}$$

Since the $a_i$ sum to 1, simplices are the natural geometry of probability distributions. Indeed, simplices have been found in the latent space of transformers, where they represent the belief-state of a hidden Markov model Shai et al. [2025].

We consider the simplicial complex: a collection of simplices 'glued' along shared vertices. It is a common object in algebraic topology Hatcher [2002] and combinatorics. The gluing allows approximations of arbitrary smooth manifolds by a process called triangulation.

Each individual simplex $\sigma \in \mathcal{F}$ is a subset of features that co-activate together. To sample $x \in \mathbb{R}^N$, we uniformly choose a face $\sigma \in \mathcal{F}$ and fire it. To fire a face, a vector $a \sim \text{Dirichlet}(1, \ldots, 1)$ is drawn over its vertices and added to $x$.

For the benchmark we chose the dimension of each simplex to be 7. We randomly draw 648 simplices, ensuring that each node is associated to at least one vertex.

**Deep Hierarchy** This distribution is a possible generalization of hierarchical pairs, where depth is more emphasized, and multiple parents may exist.

Features are arranged as nodes in a directed acyclic graph (DAG) with edges generated by an Erdős-Rényi process over the upper triangle: edge $i \to j$ (for $i < j$) exists independently with probability $p_e$.

Exactly one starting node is selected according to probabilities $p_a$ (uniform by default) and activated with value $v_0 \sim \text{Uniform}(0, 1)$. From the starting node, a random walk proceeds upward: at each step one parent is chosen uniformly at random and activated with a decayed value

$$v_{\text{parent}} = \beta\, v_{\text{child}} + (1 - \beta)\, v_{\text{child}}\, U, \tag{4}$$

where $U \sim \text{Uniform}(0, 1)$ and $\beta \in [0, 1]$ controls the decay. The walk terminates upon reaching a root node (one with no parents). All non-visited nodes remain inactive.

Each possible directed edge is drawn with probability $p_e = 18/N$. We have $\beta = 0.8$. Finally, we run the process twice and add the result together to achieve $E[L^0_{\text{True}}] \approx 8$.

**Spherical Distribution** This approach is taken from Olah and Batson [2023]. Various families of features to live on manifolds. For instance Tiblias et al. [2026] discovers various features along manifolds in LLM residual streams. See also Engels et al. [2025].

$N$ feature positions are placed approximately uniformly on a unit sphere $S^d$ (equal angular spacing for $d = 1$, Fibonacci lattice for $d = 2$, random Gaussian projection otherwise). To generate a sample, a random direction $\boldsymbol{\theta}$ is drawn uniformly on $S^d$ and a magnitude $m \sim \text{Uniform}(m_{\text{lo}}, m_{\text{hi}})$. The feature at position $\mathbf{v}_\varphi$ activates as

$$x_\varphi = m \cdot \cos\left(\frac{\alpha}{\ell}\right) \cdot \mathbf{1}\left[\frac{\alpha}{\ell} \leq \frac{\pi}{2}\right], \tag{5}$$

where $\alpha = \arccos(\mathbf{v}_\varphi \cdot \boldsymbol{\theta})$ is the geodesic distance and $\ell$ is a length-scale parameter controlling the width of the cosine bump. Smaller $\ell$ yields sparser activations concentrated near $\boldsymbol{\theta}$.

For our benchmark we set $\ell = 0.276$, $d = 4$, and $(m_{\text{lo}}, m_{\text{hi}}) = (0.5, 1.0)$.

**Torus Distribution** The Torus distribution intends to probe SAE performance on more complex feature manifolds.

$N$ feature positions are placed on a uniform grid on the flat torus $T^d = S^1 \times \cdots \times S^1$. For $d > 1$ the number of features must be a perfect $d$-th power (e.g. $N = 16$ on $T^2$ gives a $4 \times 4$ grid). To generate a sample, a random point $\boldsymbol{\theta} \in [0, 2\pi)^d$ is drawn uniformly and a magnitude $m \sim \text{Uniform}(m_{\text{lo}}, m_{\text{hi}})$. The feature at angular position $\boldsymbol{\varphi}$ activates as

$$x_\varphi = m \cdot \cos\left(\frac{d(\boldsymbol{\theta}, \boldsymbol{\varphi})}{\ell}\right) \cdot \mathbf{1}\left[\frac{d(\boldsymbol{\theta}, \boldsymbol{\varphi})}{\ell} \leq \frac{\pi}{2}\right], \tag{6}$$

where $d(\boldsymbol{\theta}, \boldsymbol{\varphi}) = \|\mathrm{wrap}(\boldsymbol{\theta} - \boldsymbol{\varphi})\|_2$ is the geodesic distance on the flat torus (each angular difference wrapped to $[0, \pi]$) and $\ell$ is the length-scale. As with the spherical distribution, smaller $\ell$ yields sparser activations.

For our benchmark we set $\ell = 0.752$, $d = 4$, and $(m_{\text{lo}}, m_{\text{hi}}) = (0.5, 1.0)$.

## B A note on MCC matching

Our MCC score differs from Song et al. [2025], Chanin and Garriga-Alonso [2026] in that we omit the absolute value around the cosine similarity term. Prior work includes it because traditional dictionary learning methods allow negative sparse codes. Since our sparse codes are constrained to be positive, the absolute value is unnecessary.

The MCC is used to match dictionary elements to features, so this choice could in principle affect all downstream metrics. In practice, Table 4 shows that the two matchings yield nearly identical $F_1$ and encoder precision (EP) scores, with a slight edge for the non-absolute variant.

EP is defined as follows. Let

$$O := \mathrm{ReLU}(V(W - b_d) + b_e), \qquad \text{EP} := \frac{\sum_{(i,j)\in M} O_{i,j}}{\sum_{i,j} O_{i,j}}, \tag{7}$$

where $O$ is the recovered one-hot fired matrix, and EP represents the ability to cleanly recover one-hot fired features.

| Matching | MCC↑ | \|MCC\| ↑ | $F_1$ ↑ | EP↑ |
|---|---|---|---|---|
| MCC | **0.753** | 0.753 | **0.302** | **0.531** |
| \|MCC\| | 0.657 | **0.759** | 0.299 | 0.528 |

Table 4: Downstream metrics are largely unaffected by the choice of matching. Metrics are calculated from a trained autoencoder on a sparse uniform distribution with $N$=1296 features, $d$=100 hidden dimensions, and SAE width 648.

We find similar slight improvements in $F_1$ scores across feature distributions and SAEs.

## C Ablations for Trained vs Constructed Autoencoders

**Autoencoder Geometry** There are meaningful differences in the geometry of the autoencoders based on which constraints you enforce. We train 4 autoencoders on the Zipfian distribution.

1. **Trained AE** is completely free to adapt, as in Equation 1.
2. **Unit-norm AE** is forced to keep the columns of $W$ unit-norm.
3. **Scalar bias AE** can freely adapt $W$, but the bias $b$ must be constant for all features.
4. **Constructed AE** has frozen $W$, but can freely adapt the bias $b$.

**Feature Norm Growth** We claim that the shape of the feature norms in Figure 7 is mediated by the bias. To test this claim we consider two ablations. The first is simply the constructed autoencoder that we have already considered. For the second we force the bias of the autoencoder to be a scalar, i.e. constant across features.

In the constructed case, with $W$ fixed, only the bias is free to adapt, yet $b$ converges to a similar profile as in the trained case (Figure 7). Furthermore, when the bias is forced to be a constant, then the feature norms slope downwards. This suggests the bias is the primary mechanism by which the autoencoder suppresses rare features.

The increasing feature norms for rarer features thus is a mechanism to offset the bias. In particular, we find that $\|W_i\|^2 + b_i \approx 1$, which helps large activations to be reconstructed more faithfully.

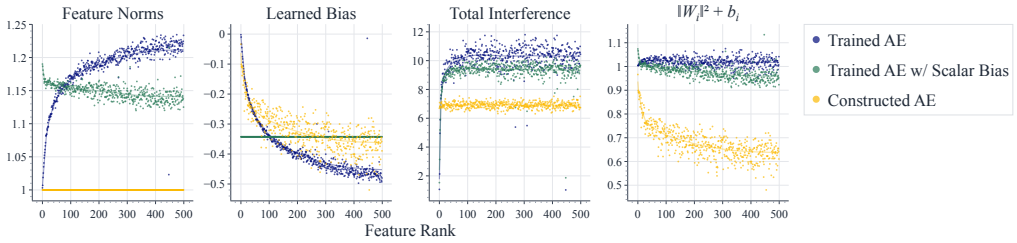



Figure 7: The feature norm for the autoencoder with scalar-constrained bias is decreasing rather than increasing, unlike the other autoencoder variants.

## D Autoencoder constraints

A natural design choice to consider when training the toy autoencoder is whether to constrain the columns of $W$ to unit norm, matching the convention used by the downstream SAE. We choose not to.

**The case for unit norm.** The unit-norm constraint places the autoencoder and SAE on the same footing, so that any trained-versus-constructed gap must come from angular arrangement alone rather than norm reallocation. We ran this ablation: the main result of Section 5 survives under $\|W_i\| = 1$, confirming that norm reallocation is not driving the gap.

**Why we default to free norms.** Free-norm autoencoders achieve strictly lower reconstruction MSE. The unit-norm constraint costs the autoencoder compression capacity, and since the purpose of the toy autoencoder is to produce an optimally (linearly) compressed representation against which SAEs must recover features, free norms give the more informative test.

Furthermore, any regularity baked into the ground truth is, by construction, a regularity downstream SAEs can exploit. This could limit ToyBench as a test-bed for SAE design. We are not aware of direct measurements of per-feature effective norms in real transformer residual streams, so we cannot say which convention would be more correct, thus Occam's razor tells us to go for the simpler option.

**Summary.** Our default reflects a prior that heterogeneous norms are more likely to be present than absent. We believe that properties one cannot rule out should be preserved rather than stripped. Finally, based on the results in Figure 5 we can see that the difference is negligible.

## E On correlations and interference

Across all distributions studied in this paper, we observe a positive relationship between the empirical correlation of features and the interference of their learned representation in the trained autoencoder. Table 4 shows this relationship for each distribution.

This is intuitive: correlated features are likelier to fire together, so the model incurs less loss from placing them in higher interference configurations. The sparse uniform distribution is the exception, as it has no correlations between features by construction, and accordingly the measured trend is zero.

## F Additional Benchmark Details