Unstable Features, Reproducible Subspaces: Understanding Seed Dependence in Sparse Autoencoders

Anonymous Authors[1]

Abstract

Sparse autoencoders (SAEs) are widely used to interpret neural network representations, but their utility depends on whether the learned features are reproducible across training runs. We study this question through *feature stability*: for each SAE feature, we estimate the probability that a similar feature reappears in an independently trained SAE. This yields a scalable per-feature signal that separates stable from unstable features. In a large-scale study across seeds, models, layers, dictionary sizes, and SAE variants, we find a pronounced functional asymmetry: stable features carry most of the reconstruction- and prediction-relevant signal, while unstable features have weak marginal impact and are dominated by low-frequency surface-form triggers in both activation statistics and automatic explanations. Geometrically, unstable features are individually non-reproducible but concentrate in reproducible lower-rank subspaces, suggesting that seed dependence often reflects basis ambiguity within a shared region of activation space rather than pure noise. A controlled synthetic model makes this mechanism explicit, showing that low-rank ground-truth features can be recovered at the subspace level while remaining non-identifiable as individual SAE latents across seeds. Finally, by pooling unique cross-seed features, we construct SAEs from stable features and find no evidence of a stability–explained-variance trade-off in this setting. Together, these results support cross-seed stability as a practical faithfulness filter and null check for SAE-based interpretability.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. Introduction

Sparse autoencoders (SAEs) are a central tool in mechanistic interpretability because they aim to decompose model activations into sparse, human-interpretable features (Bricken et al., 2023; Cunningham et al., 2023; Gao et al., 2025). A basic question, however, is whether these features are reproducible: if SAEs trained on the same activations but with different random seeds learn substantially different dictionaries, then single-run interpretations become difficult to trust (Paulo & Belrose, 2025a; Leask et al., 2025a).

We study this problem through *feature stability*: the probability that a feature reappears in an independently trained SAE under a cosine-similarity matching rule. This gives a scalable per-feature signal that helps separate reproducible directions in activation space from seed-specific artifacts. Stability is not sufficient for faithfulness, but it is a useful filter and a stringent null check, especially given that automatic interpretation pipelines can assign plausible explanations even in settings where the underlying representations are random or non-canonical (Heap et al., 2025; Bhalla et al., 2026).

Using this signal, we compare stable and unstable features by their functional impact, token structure, and automatic explanations; analyze their geometry in decoder space; probe a low-rank mechanism in a controlled synthetic model; and test whether stable features pooled across runs can be used to construct seed-robust SAEs. We also compare pretrained-model SAEs against random-model controls, turning seed dependence from a nuisance into a measurable property that can be used both to evaluate interpretability claims and to construct more reproducible dictionaries.

Our contributions are as follows.

- **Functional asymmetry of stable and unstable features.** Stable features carry most of the reconstruction- and prediction-relevant signal, whereas unstable features have weak marginal impact and are dominated by low-frequency surface-form triggers in both activation statistics and automatic explanations.
- **Subspace recovery without feature identifiability.** Decoder-space analysis shows that unstable features

are individually non-reproducible but collectively span reproducible lower-rank subspaces; in a controlled synthetic setting, low-rank ground-truth features are likewise recovered at the subspace level while failing to align with individual SAE latents.

- **No stability–EV trade-off in feature-pool construction.** Using a deduplicated pool of unique features from multiple runs, we construct SAEs from stable cross-seed features and find that higher explained variance coincides with higher mean feature probability after tuning, suggesting that stability and reconstruction quality need not trade off in this construction setting.

2. Related Work

Feature stability. Paulo & Belrose (2025a) showed that SAEs trained on the same model and data, differing only in random seed, can learn substantially different feature sets. Leask et al. (2025a) argued that SAE latents are not canonical units of analysis, showing that features vary with dictionary size and can be merged across dictionaries. Fel et al. (2025) mitigate instability by anchoring dictionary atoms to data-derived archetypes. Gadgil et al. (2025) propose ensembling independently initialized SAEs via naive bagging and boosting. Our construction of an SAE from unique cross-seed features follows the same bagging intuition, but deduplicates the pooled feature set and selects a fixed-size subset of high-stability features rather than simply concatenating all latents from all runs. Chen et al. (2025) improve consistency by assigning features to activation-frequency groups and adapting encoder biases to enforce group-specific target frequencies. Cho et al. (2025) argue that external training data can be partially OOD for the base model and report that training on model-generated data improves cross-seed stability. Wang et al. (2025b) impose an ordered latent structure in which increasingly large prefixes of the feature set reconstruct the activation, reducing permutation non-identifiability and improving consistency. Heap et al. (2025) showed that SAEs trained on randomly initialized transformers can still yield plausible automatic interpretations, motivating random-model controls for SAE interpretability. Bhalla et al. (2026) give a complementary geometric account, arguing that multidimensional concepts can admit multiple valid SAE bases, making seed-dependent decompositions expected.

In this work, we estimate per-feature reappearance probabilities to study instability through seed-dependent basis choices within reproducible lower-rank decoder subspaces, probing this view with a controlled low-rank synthetic model and comparing stable and unstable features by reconstruction impact, downstream effects, and automatic explanations.

Finding similar features across SAEs. Balcells et al. (2024) track how individual SAE features evolve between adjacent layers, while Balagansky et al. (2025b) introduce a data-free method for matching SAE features across layers; Laptev et al. (2025) extend this line by constructing cross-layer feature-flow graphs. Wang et al. (2025a) identify corresponding SAE features across different language-model architectures. Anonymous Authors (2026) bridge the stability and matching literatures by benchmarking cross-seed SAE correspondence methods, including cosine and optimal-transport matchers, and evaluating them functionally via ablation effects and substitution tests.

3. Preliminaries

3.1. Sparse Autoencoders

A sparse autoencoder (SAE) is designed to represent hidden states as a sparse linear combination of feature embeddings (Bricken et al., 2023; Cunningham et al., 2023); each coefficient is treated as an activation magnitude, or importance of this feature, which are assumed to be non-negative.

Architecturally, an SAE has the following form:

$$\begin{aligned} \boldsymbol{z} &= \sigma(\boldsymbol{W}_{\text{enc}}\boldsymbol{h} + \boldsymbol{b}_{\text{enc}}) \in \mathbb{R}^F, \\ \hat{\boldsymbol{h}} &= \boldsymbol{W}_{\text{dec}}\boldsymbol{z} + \boldsymbol{b}_{\text{dec}} \in \mathbb{R}^d, \end{aligned} \tag{1}$$

where $\boldsymbol{h}$ is a hidden state vector, $\boldsymbol{z}$ is a vector of feature activation magnitudes, and σ is a nonlinear activation function that induces sparsity and non-negativity. SAEs are trained on reconstruction loss plus optional regularization loss that controls sparsity level scaled by coefficient $\alpha \geq 0$:

$$\mathcal{L} = \|\boldsymbol{h} - \hat{\boldsymbol{h}}\|_2^2 + \alpha\mathcal{L}_{\text{reg}}(\boldsymbol{z}). \tag{2}$$

Vanilla SAE (Cunningham et al., 2023) simply uses ReLU as nonlinearity and $\mathcal{L}_{\text{reg}} = \|\boldsymbol{z}\|_1$. Widely used *TopK SAE* (Gao et al., 2025) imposes sparsity on $\boldsymbol{z}$ by zeroing out all entries whose values are outside the top-k elements, and *Batch TopK SAE* (Bussmann et al., 2024) operates on the batch level instead of single-entry and zeroes out elements smaller than top-(batch size $\times$ k); these activations allow to fix the desired sparsity level and drop the $\mathcal{L}_{\text{reg}}$ from loss function. *JumpReLU SAE* (Rajamanoharan et al., 2024) modifies Vanilla SAE by introducing learnable threshold parameter $\theta \in \mathbb{R}^F$ that independently controls the minimal activation value for each feature, and replacing nonlinearity with $\mathrm{JumpReLU}(\boldsymbol{x}) = \boldsymbol{x}\mathrm{H}(\boldsymbol{x} - \theta)$, where H is the Heaviside function.

We denote $\boldsymbol{f}_i$ as the entity corresponding to the feature with index i, and $\boldsymbol{z}_i$ as the activation value of that feature. Its embedding $\boldsymbol{e}_i \in \mathbb{R}^d$ is the ith column of the decoder matrix, and we assume that all embeddings lie on a unit sphere (see Appendix A for details on normalization procedure).

3.2. Feature Matching and Stability

Two SAEs trained on similar hidden state distribution (e.g., on the same layer) are expected to converge towards similar representations of the hidden states and therefore similar feature representations. Since SAEs are invariant to feature ordering and two different initializations might produce same dictionaries, but with different indices (Balagansky et al., 2025a), a linear assignment problem is usually solved (Paulo & Belrose, 2025b; Balagansky et al., 2025a; Fel et al., 2025): given two dictionaries $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$, we seek for a permutation matrix $\boldsymbol{\Pi}$ so that discrepancy between dictionaries $\boldsymbol{D}_1$ and $\boldsymbol{\Pi}\boldsymbol{D}_2$ is minimized; this gives a one-to-one correspondence. The quality of resulting matching is a measure of SAE stability - if it is low, then features are mostly different. This is primary method for previous works on the problem we investigate in this paper.

In this work we adopt a many-to-one alternative: for each element from $\boldsymbol{D}_1$, we find the most similar element from $\boldsymbol{D}_2$, measured by maximum cosine similarity. This approach is much less computationally expensive and allows to compute stability of individual features, helping us understand how their individual properties correspond to the problem of instability.

4. Methodology

We study how reliably individual SAE features reappear across random initializations. Throughout, we represent each feature by its decoder vector, and ℓ_2-normalize all decoder columns so that cosine similarity reduces to a dot product.

Feature matching. Given two SAEs A and B with decoder columns $\{\boldsymbol{e}_i^{(A)}\}_{i=1}^F$ and $\{\boldsymbol{e}_j^{(B)}\}_{j=1}^F$, we say that features i and j *match* if $\cos\big(\boldsymbol{e}_i^{(A)}, \boldsymbol{e}_j^{(B)}\big) \geq \theta$, following (Leask et al., 2025b), $\theta = 0.7$ unless stated otherwise.

Reappearance probability. We train $N+1$ SAEs on identical data and hyperparameters with different random seeds, and choose one run as an anchor ($k = 0$). For each anchor feature embedding $\boldsymbol{e}_i^{(0)}$, we count how many of the other N SAEs contain a feature with cosine similarity at least θ:

$$X_{0,i} = \sum_{k=1}^{N} \mathbf{1}\left\{ \max_{j \in \{1,\dots,F\}} \cos\big(\boldsymbol{e}_i^{(0)}, \boldsymbol{e}_j^{(k)}\big) \geq \theta \right\}. \quad (3)$$

Let

$$p_i := \Pr\left(\max_j \cos(\boldsymbol{e}_i^{(0)}, \boldsymbol{e}_j^{(k)}) \geq \theta \right), \qquad k \neq 0.$$

Assuming independent seeds, $X_{0,i} \sim \mathrm{Binomial}(N, p_i)$, and we estimate p_i by

$$\hat{p}_i = \frac{X_{0,i}}{N}. \quad (4)$$

Across features, we use the empirical CDF $\hat{\mathbf{F}}(p) = \frac{1}{F}\sum_{i=1}^{F} \mathbf{1}\{\hat{p}_i \leq p\}$, which converges uniformly to $\mathbf{F}(p)$ (Glivenko–Cantelli; (Glivenko, 1933; Cantelli, 1933)).

Endpoint stability. We focus on *endpoint* behavior: features that reappear in almost none or almost all runs. Fixing $\varepsilon = 0.05$, we call a feature *unstable* if $\hat{p}(\boldsymbol{f}_i^{(0)}) \leq \varepsilon$ and *stable* if $\hat{p}(\boldsymbol{f}_i^{(0)}) \geq 1 - \varepsilon$. Our main task is therefore distilled into finding $\mathbf{F}(\varepsilon)$ and $\mathbf{F}(1-\varepsilon)$, via empirical distribution function.

Unique feature pool. For convenience, we can also introduce a Unique feature pool, which consists of pairs of SAE id S and feature index i,

$$\begin{aligned} \mathbb{U} \subseteq \{(S,i)\} \text{ s.t. } \quad & \forall\, (S_1, i_1) \neq (S_2, i_2) \in \mathbb{U} : \\ & \cos\big(\boldsymbol{e}_{i_1}^{(S_1)},\, \boldsymbol{e}_{i_2}^{(S_2)}\big) < \theta. \end{aligned} \quad (5)$$

This set of features allows us to construct new SAEs from this pool, but with different feature probability distribution. See Section 5.3.

5. Quantitative Analysis

Experimental Setup. Unless stated otherwise, we report results in a single, well-instrumented configuration: we train 96 TopK SAEs (so $N = 95$ comparisons per anchor) on GPT-2 activations at the residual stream on layer 7, with TopK $= 64$ and dictionary size $F = 2^{14}$. Figure 1 summarizes the resulting empirical reappearance rates $\hat{p}$. The distribution is sharply concentrated near the endpoints: a large mass of features is recovered in nearly every run, while a smaller but clearly separated group is recovered in nearly none; the remaining features occupy a comparatively thin middle region (mean $\hat{p} \approx 0.75$ in this setting). Appendix B.2 (Figure 10a) provides a sensitivity check: as the cosine threshold θ is varied, the stable fraction decreases and the unstable fraction increases, and the overall stability profile depends strongly on the SAE nonlinearity (with TopK and BatchTopK nearly overlapping, JumpReLU intermediate, and Vanilla exhibiting the strongest endpoint stability).

We now use stability to compare *what unstable vs. stable features look like* and *how much they matter*. Concretely, we show that: (i) unstable features activate less frequently and (on average) with smaller magnitude tails (as quantified in Section 5.1 and shown in Appendix B.3, Figure 10b); (ii) unstable features concentrate on lower-level lexical triggers with lower token diversity (Figure 2); and (iii) even after aggressively masking them, unstable features have much smaller impact on reconstruction and next-token loss than stable features under a frequency-matched protocol (Figure 3).

Throughout, we use the partitions from Section 4 at $\varepsilon =$

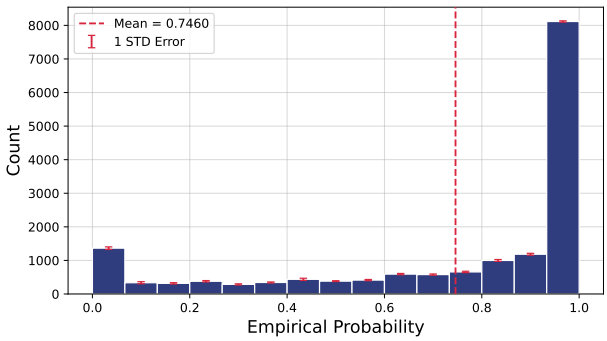


Figure 1. Feature reappearance across seeds in the main setup (See Section 5). Empirical distribution of reappearance rates $\hat{p}$ (Eq. 4) for anchor features; the vertical dashed line marks the mean, and error bars indicate variability across 5 anchor choices.

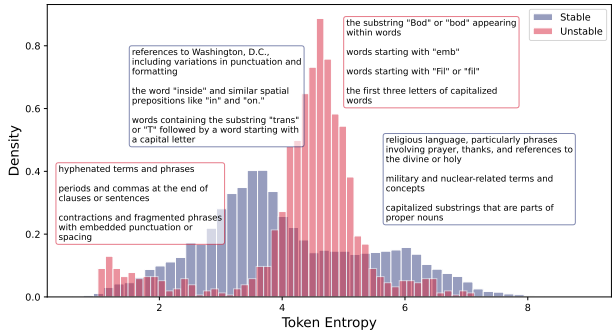


Figure 2. **Token diversity for stable vs. unstable features.** Token entropy H_i with representative feature interpretations.

0.05:

$$\mathcal{U}_\varepsilon = \{i \mid \hat{p}(\boldsymbol{f}_i^{(0)}) \le \varepsilon\}, \qquad \mathcal{S}_\varepsilon = \{i \mid \hat{p}(\boldsymbol{f}_i^{(0)}) \ge 1-\varepsilon\}.$$

All statistics below are computed on a held-out token collection; full evaluation and sampling details are deferred to Appendix B.

5.1. Activation Statistics and Token Structure

Activation frequency and magnitude. We summarize feature usage with activation frequency ω_i (fraction of evaluated token positions where feature i activates) and conditional mean magnitude μ_i (mean activation value conditioned on activation); formal definitions are in Appendix B.3. Appendix B.3 (Figure 10b) shows that unstable features activate less frequently on average: their mean frequency is $\overline{\omega}_{\mathcal{U}_\varepsilon} \approx 0.0018$ (0.18%), compared to $\overline{\omega}_{\mathcal{S}_\varepsilon} \approx 0.0044$ (0.44%) for stable features. Stable features also exhibit a heavier high-magnitude tail in μ_i. (We defer dataset, batching, and exact N_{tok} details to Appendix B.1.)

Token entropy (token diversity). We next measure how diverse the *token identities* are among positions where a feature activates. For each feature i, we compute the entropy H_i of the empirical distribution over vocabulary token IDs restricted to positions with $z_{n,i} > 0$ (formal definition and implementation details in Appendix B.4). Intuitively, low H_i indicates activation concentrated on a small set of tokens, while high H_i indicates broader lexical support.

Crucially, *both* stable and unstable features exhibit two dominant entropy regimes, but with different content. For unstable features, the lowest-entropy mode is dominated by punctuation/formatting triggers, while a higher-entropy mode corresponds to short subword fragments and brittle substrings that match across many tokens. For stable features, a lower-entropy mode (typically higher than the unstable punctuation regime) corresponds to single words and tight synonym clusters, while the highest-entropy mode corresponds to higher-level concepts with many lexical realizations. Representative interpretations for these regimes are shown as callouts in Figure 2.

Automatic interpretability. These token-diversity differences are mirrored by SAEBench auto-interpretation results in Appendix D. Stable features achieve higher detection scores overall (Figure 15b); in particular, the fraction of stable features with perfect detection score 1 is approximately $4.5\times$ larger than for unstable features. Generated explanations also differ systematically: unstable features are more often described via surface-form triggers such as substrings, capitalization, and punctuation, whereas stable features are more often described via phrases, constructions, syntactic roles, and broader semantic groupings (Table 2). The same split appears in keyword frequencies over explanation text: `substring` appears in 38.9% of unstable explanations versus 11.3% of stable ones, while `phrase` rises from 4.1% to 32.0%. As an auxiliary test, GPT-5 predicts stable vs. unstable from explanation text alone with 0.88 accuracy (Appendix D.6).

5.2. Impact on Reconstruction and Next-Token Loss

Frequency-matched masking protocol. To control for the lower activation frequency of unstable features (Section 5.1), for each N we mask N features sampled uniformly at random from $\mathcal{S}_\varepsilon$ and $4N$ features sampled uniformly at random from $\mathcal{U}_\varepsilon$; full sampling, uncertainty, and reweighting details are deferred to Appendix B.5.

Explained variance (EV). We measure how much reconstruction quality degrades after masking selected features by evaluating explained variance on the resulting SAE reconstructions (formal definition in Appendix B.5). Figure 3 (left) shows that masking unstable features changes EV only slightly even when masking $4N$ of them, whereas masking far fewer stable features yields a substantially larger EV drop. This indicates that stable features account for most of the reconstruction-relevant variance.

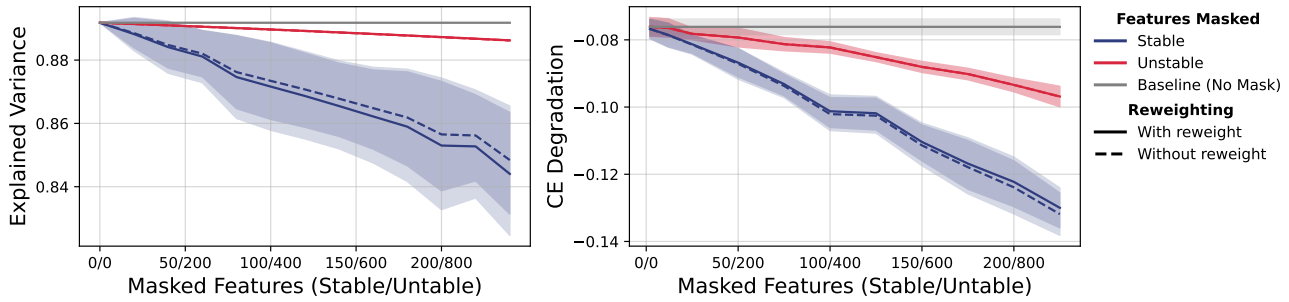


Figure 3. **Impact of masking stable vs. unstable features on reconstruction and next-token loss. Left**: explained variance (EV) under feature masking. **Right**: change in next-token loss under activation patching with masked-feature reconstructions. We mask N stable features and $4N$ unstable features to approximately match expected active mass. Solid vs. dashed curves correspond to using reweighting vs. not using reweighting; full protocol details are deferred to Appendix B.5.

Next-token loss under activation patching. To quantify downstream impact on the base model, we patch the residual stream at the SAE training location with the masked-feature reconstruction and measure the resulting change in next-token cross-entropy, $\mathrm{CE}_{\text{base}} - \mathrm{CE}_{\text{patched}}$ (details in Appendix B.6). As shown in Figure 3 (right), even removing a large number of unstable features produces only a modest loss change, while masking far fewer stable features yields substantially larger degradation.

Overall, unstable features behave like low-frequency lexical artifacts with limited functional impact, whereas stable features capture most of the reconstruction- and prediction-relevant structure.

5.3. Constructing SAE from Unique Features

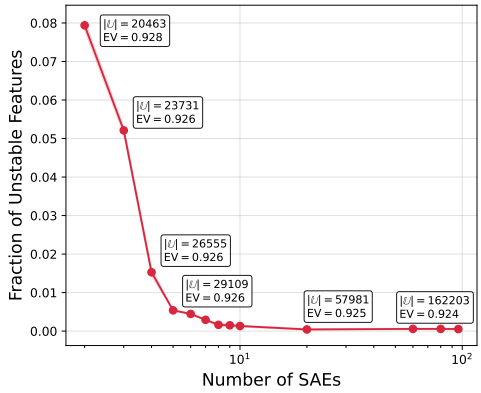


Figure 4. **Unstable-feature fraction in the most-probable construction as a function of the number of source SAEs.** As the source pool grows, the selected dictionary contains progressively fewer unstable features and is already dominated by stable ones for modest ensemble sizes. Boxes next to selected points report both the size of the pooled unique-feature set and the explained variance of the constructed SAE for that source-pool size.

To test whether stable features can be turned into a practical initialization, we train a collection of independently seeded SAEs and merge their decoder features into a unique pool $\mathbb{U}$ using Eq. 5, so that near-duplicate cross-seed features appear only once. For any chosen source-pool size n, we form the corresponding pool, estimate the empirical reappearance probability $\hat{p}$ of each pooled feature within that subset, and then construct new SAEs with $F = 16{,}384$ latents in three ways: using the F most probable features, the F least probable features, or an *equiprobable* baseline that samples pooled features uniformly regardless of $\hat{p}$. We then briefly tune each constructed SAE. Appendix Sections B.7 and B.8 gives the full tuning curves and the probability histograms of the selected features.

This construction shows that only a modest number of source SAEs is needed to recover a high-quality stable dictionary. As shown in Figure 4, the most-probable construction quickly eliminates unstable features as n grows and is already dominated by stable ones for modest source pools. The detailed EV curves in Appendix Section B.7 show that SAEs initialized from the most probable features recover to near-baseline explained variance after brief tuning, while the least-probable construction remains much worse and the equiprobable baseline lies in between. The probability histograms in Appendix Section B.8 make this mechanism explicit: increasing n pushes the most-probable construction toward near-certain reappearing features, whereas the least-probable construction concentrates on seed-specific ones.

Overall, most of the reusable cross-seed structure is recovered early. Adding many more SAEs still refines the pool, but a large ensemble is not necessary to construct a seed-robust SAE with essentially no explained-variance drop after tuning.

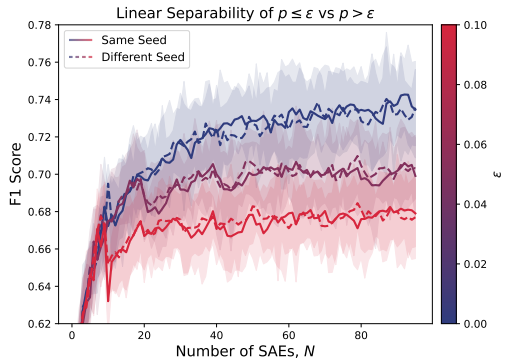


Figure 5. **F1 score for classification separating unstable features from all others.** Solid curves report within-seed evaluation, while dashed curves report transfer of the classifier to a different seed.

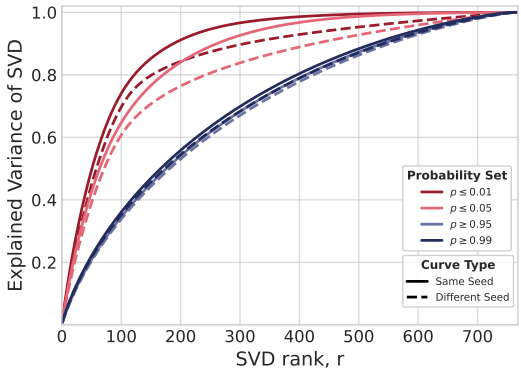


Figure 6. **Explained variance of singular values of decoder submatrices versus SVD rank: within-seed (solid) and cross-seed transfer (dashed)**. This indicates that the subspaces learned in one seed accurately approximate those in other seeds.

6. Geometric Analysis

We now investigate features geometrically at the level of decoder vectors $e_j^{(s)} \in \mathbb{R}^d$. Our goal is to show that: (i) unstable features are approximately linearly separable from the rest in decoder space, (ii) unstable features concentrate in a lower effective-dimensional subspace than stable features, and (iii) the *subspace* spanned by unstable features exhibits a high degree of cross-seed reproducibility.

6.1. Linear Separability in Decoder Space

We first ask whether the decoder vectors themselves carry a geometric signature of instability. As early, for each seed s let $\hat{p}(\boldsymbol{f}_i^{(s)})$ denote the probability of feature i and define the set of unstable features as $\mathcal{U}_\varepsilon^{(s)} = \{i : \hat{p}(\boldsymbol{f}_i^{(s)}) \le \varepsilon\}$.

We train a logistic regression classifier to predict the binary label $y_i^{(s)} = \mathbf{1}\{i \in \mathcal{U}_\varepsilon^{(s)}\}$ from the corresponding decoder vector $e_i^{(s)} \in \mathbb{R}^d$. In addition to within-seed evaluation, we test transferability by applying a classifier trained on one seed to the decoder vectors of another seed.

We evaluate performance using the F1 score (Figure 5) as a function of N, the number of independently trained SAEs used to estimate the probabilities $\hat{p}(\boldsymbol{f}_j^{(s)})$. We find that the within-seed and cross-seed F1 scores are very close, and for sufficiently large N they saturate at approximately 0.73 for $\varepsilon = 0$ and 0.67 for $\varepsilon = 0.1$.

6.2. Effective Rank of Features Subspaces

We next compare the dimensionality of the decoder vector subspaces associated with stable and unstable features. Given an index set $\mathcal{I}_\varepsilon^{(s)}$ (either $\mathcal{U}_\varepsilon^{(s)}$ for unstable features or $\mathcal{S}_\varepsilon^{(s)}$ for stable features), define the submatrix of the decoder

$$X_{\mathcal{I}_\varepsilon}^{(s)} := \boldsymbol{W}_{\text{dec}}^{(s)}[:, \mathcal{I}_\varepsilon^{(s)}] \in \mathbb{R}^{d \times m_s}, \qquad m_s := |\mathcal{I}_\varepsilon^{(s)}|.$$

We quantify dimensionality via the effective rank (ER) (formal definition in Appendix E). Low effective rank suggests that most variation across the features is explained by a small number of dominant concepts, reflecting low semantic diversity.

We compute effective ranks averaged over $N = 96$ seeds and find that unstable features consistently exhibit smaller effective rank than stable features across seeds (see Table 5 and implementation details in Appendix E).

6.3. Overlap of Subspaces

We show that unstable decoder vectors are (i) approximately linearly separable from the rest and (ii) concentrated in a low effective-dimensional subspace. We now turn to investigating whether the subspace of unstable features is reproducible across seeds.

We consider the explained variance $\text{EV}_{\text{SVD}}^{(s)}(r)$ of singular values of matrix $X_{\mathcal{I}_\varepsilon}^{(s)}$ (formal definition in Appendix E), which measures how well the top-r singular subspace captures the variance within the features set $\mathcal{I}_\varepsilon$.

Using the SVD of $X_{\mathcal{I}_\varepsilon}^{(s)}$ and the subspace interpretation of $\text{EV}_{\text{SVD}}^{(s)}(r)$ (for details, see Appendix E), we define the cross-seed analogue $\text{EV}_{\text{SVD}}^{(a\to s)}(r)$ for $s \ne a$, which measures how well the top-r singular subspace from seed a explains the feature subspace in seed s.

Figure 6 compares $\text{EV}_{\text{SVD}}^{(s)}(r)$ to $\text{EV}_{\text{SVD}}^{(a\to s)}(r)$, averaged over all choices of a and s among $N = 96$ seeds, for $\mathcal{U}_\varepsilon$ and $\mathcal{S}_\varepsilon$ with $\varepsilon \in \{0.01, 0.05\}$. We find that the top-$r$ singular subspace learned in one seed provides an accurate approximation to the corresponding feature subspaces in other seeds, including for unstable features.

Taken together, although individual unstable features rarely reappear across seeds, the low-rank subspaces spanned by them are reproducible. This suggests that feature instabil-

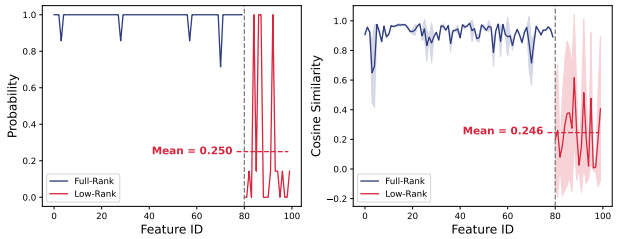


Figure 7. **Synthetic low-rank model ($d = 32$, $r = 2$, $k = 8$).** The first 80 ground-truth features are full-rank features and the last 20 lie in a shared rank-2 subspace. Full-rank features (blue) have near-perfect cross-seed reappearance probability and cosine similarity to their matched ground-truth features, whereas low-rank features (red) do not.

ity is primarily driven by seed-dependent mixing within a shared subspace, rather than changes in the subspace itself.

7. A Controlled Low-Rank Synthetic Model

To test whether shared low-rank structure alone can generate this pattern, we study a synthetic dictionary in which only a subset of the ground-truth features is constrained to a common low-dimensional subspace. Concretely, we define

$$W = \begin{bmatrix} D \\ UV \end{bmatrix}, \qquad D \in \mathbb{R}^{n_{\text{full}} \times d}, \qquad U \in \mathbb{R}^{n_{\text{low}} \times r},$$

$$V \in \mathbb{R}^{r \times d}, \qquad r < d,$$

where the rows of D are generic full-rank features and the rows of UV lie in the shared subspace $L = \text{rowspan}(V) \subset \mathbb{R}^d$. We generate synthetic activations by sampling k rows of W uniformly without replacement, summing them with unit coefficients, and training TopK SAEs from multiple random seeds to reconstruct the resulting vectors.

This setup isolates a concrete mechanism for instability. If the full-rank features are identifiable as individual directions, they should be recovered consistently across seeds. By contrast, if the low-rank block is identifiable mainly at the level of its span, different seeds can learn different mixtures within L: individual low-rank features need not match, even though the subspace itself is reproducible.

Figure 7 confirms this prediction in the main synthetic setting ($d = 32$, $r = 2$, $k = 8$). The full-rank features are recovered stably, with reappearance probabilities close to 1 and cosine similarity to matched ground-truth features also close to 1. In contrast, the low-rank block has much lower mean reappearance probability (≈ 0.25) and substantially worse one-to-one recovery. Appendix C shows the same qualitative split in additional settings and further shows that the learned low-rank block has lower effective rank and non-random cross-seed subspace similarity.

This controlled model therefore reproduces the core geometric phenomenon from real SAEs: instability can arise from seed-dependent basis choices within a reproducible low-rank subspace, rather than from the absence of shared underlying structure.

8. Other Setups

We next test whether the stability patterns from the main setting persist under changes to the base model, layer, and dictionary size, and under alternative SAE nonlinearities, training budgets, and a random-model null baseline. Overall, these ablations show that a non-trivial unstable subset coexists with a large stable subset across settings, and that the key empirical trends in this paper are not artifacts of a single model, layer, or training run.

Across base models, layers, and dictionary sizes. To check that the stability phenomena are not confined to a single representation, we train TopK SAEs across several base models, layers, and dictionary sizes. Across all evaluated settings, we observe a substantial stable subset alongside a non-trivial unstable subset, indicating that seed dependence is a general property of this training objective rather than a one-off artifact. Stability varies with the representation: for GPT-2 and Gemma-2, instability generally decreases with depth, while some Pythia settings deviate from this trend. Full endpoint fractions for all configurations are reported in Appendix F (Table 6).

Effect of sparsity mechanism and reconstruction quality. Table 1 compares SAE variants on GPT-2 layer 7 at fixed $F = 2^{14}$, including EV. TopK and BatchTopK are nearly indistinguishable both in stability and EV (≈ 0.892), suggesting that the dominant source of seed dependence is the hard k-sparsity constraint itself rather than whether it is imposed per-token or per-batch. Vanilla (ReLU+ℓ_1) is extremely stable (near-zero unstable fraction), but at comparable sparsity it has substantially lower EV than TopK, highlighting a clear stability–reconstruction tradeoff when moving along the choice of sparsity/activation mechanism. JumpReLU provides an intermediate point on this tradeoff curve: it improves stability relative to TopK while achieving EV close to TopK at higher sparsities.

More SAE training does not remove instability. Figure 8 shows the unstable fraction in the main TopK setting as a function of total SAE training tokens from 10M to 10B. The unstable fraction decreases early in training and then approaches a clear non-zero plateau ($\approx 8\%$ at 10B tokens in this setting). In contrast, the stable fraction continues to increase with additional training but with diminishing returns; we report the corresponding curve and full setup details in Appendix (Fig. 16a).

Dead-salmon baseline: trained vs. random transformers. Finally, we repeat the same SAE training procedure

Table 1. SAE type comparison on GPT-2 at layer 7 with $F = 2^{14}$.

SAE type	Sparsity	EV	Unstable	Stable
Vanilla (ReLU + ℓ_1)	62	0.833	0.003	0.810
	91	0.856	0.006	0.764
TopK	64	0.892	0.098	0.522
BatchTopK	64	0.892	0.097	0.523
JumpReLU	29	0.841	0.049	0.624
	58	0.884	0.034	0.620
	93	0.902	0.038	0.602

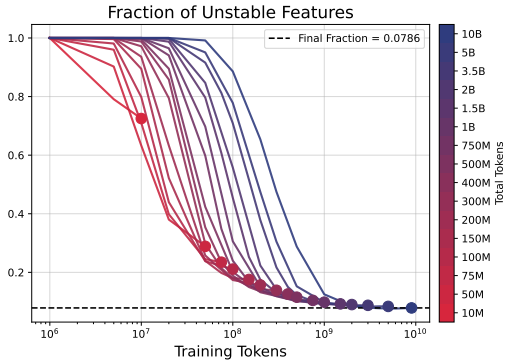


Figure 8. **Unstable fraction vs. SAE training tokens.** Fraction of unstable features in the main TopK setting as a function of total SAE training tokens.

on activations from a randomly initialized GPT-2 (same architecture and SAE hyperparameters). Our stability metric sharply distinguishes these cases: Figure 9 compares stability as a function of the cosine matching threshold θ for SAEs trained on a trained GPT-2 versus a random GPT-2. Across a wide range of θ, the trained-model SAEs retain a large stable subset, whereas in the random-model setting the stable fraction collapses and instability dominates. Importantly, automatic interpretation alone would not reliably flag this failure mode: SAEs trained on the random model can still achieve high detection scores (Appendix F.3, Fig. 16b). Together, these results motivate cross-seed recurrence as a stringent null check that helps separate reproducible directions in a trained model's activation space from seed-dependent artifacts of the SAE objective.

9. Conclusion

Our results establish feature stability, measured via cross-run reappearance probability, as a critical lens for sparse autoencoder interpretability. Stability cleanly partitions features into two regimes: a high-impact stable subset that captures structural and compositional patterns essential for reconstruction and prediction, and a low-impact unstable subset that behaves as brittle, seed-sensitive artifacts with surface-level triggers. Crucially, instability is not random noise. Unstable features concentrate in a reproducible low-

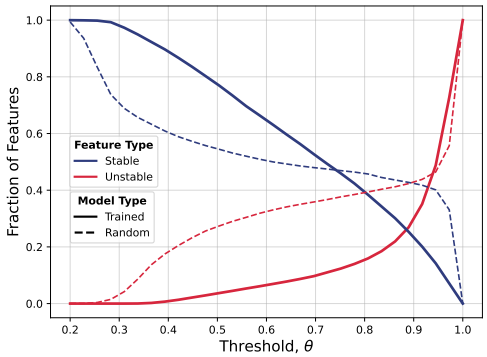


Figure 9. **Dead-salmon control: stability on trained vs. random transformers.** Fractions of stable and unstable features as a function of cosine matching threshold θ, comparing SAEs trained on a trained GPT-2 versus a randomly initialized GPT-2 (same architecture and SAE setup).

rank subspace, suggesting seed dependence often reflects arbitrary mixing within a shared region of activation space. Our controlled toy model makes this mechanism explicit, showing that low-rank ground-truth features can be recovered at the subspace level while remaining non-identifiable as individual SAE latents across seeds. This insight motivates a methodological shift. Researchers should report stability distributions rather than single-run artifacts, prioritize stable features for mechanistic claims and interventions, and treat unstable features as a distinct low-rank regime rather than interchangeable discoveries. Dead-salmon controls demonstrate that automatic interpretability pipelines can yield compelling but unstable features even in random models, underscoring that cross-seed recurrence is a necessary complement to plausibility-based validation. Stability provides a scalable, model-agnostic filter that increases the reliability of sparse autoencoder conclusions. By pooling stable features across seeds and applying brief post-training, researchers can construct seed-robust dictionaries without sacrificing reconstruction quality. While stability does not replace causal validation, it offers a practical foundation for reproducible science, enabling shareable feature dictionaries and ensuring that mechanistic insights reflect properties of the model rather than artifacts of optimization.

Impact Statement

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Anonymous Authors. Benchmarking cross-seed feature correspondence in sparse autoencoders. Under review for TMLR, OpenReview, 2026. URL https:

//openreview.net/forum?id=5cy6WtSC8f. OpenReview submission.

Balagansky, N., Maksimov, I., and Gavrilov, D. Mechanistic permutability: Match features across layers. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=MDvecs7EvO.

Balagansky, N., Maksimov, I., and Gavrilov, D. Mechanistic permutability: Match features across layers. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=MDvecs7EvO.

Balcells, D., Lerner, B., Oesterle, M., Ucar, E., and Heimersheim, S. Evolution of SAE features across layers in LLMs, 2024. URL https://arxiv.org/abs/2410.08869. Presented at the Attributing Model Behavior at Scale (ATTRIB) workshop at NeurIPS 2024.

Bhalla, U., Fel, T., Rager, C., Feucht, S., Haklay, T., Wurgaft, D., Boppana, S., Kowal, M., Shyam, V., Lewis, O., McGrath, T., Merullo, J., Geiger, A., and Lubana, E. S. Do sparse autoencoders capture concept manifolds?, 2026. URL https://arxiv.org/abs/2604.28119.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2, 2023.

Bussmann, B., Leask, P., and Nanda, N. Batchtopk sparse autoencoders, 2024. URL https://arxiv.org/abs/2412.06410.

Cantelli, F. P. Sulla determinazione empirica delle leggi di probabilità. *Giornale dell'Istituto Italiano degli Attuari*, 4:421–424, 1933.

Chen, S., Sheen, H., Xiong, X., Wang, T., and Yang, Z. Taming polysemanticity in LLMs: Provable feature recovery via sparse autoencoders, 2025. URL https://arxiv.org/abs/2506.14002.

Cho, S., Oh, H., Lee, D., Vieira, L. E. R., Bermingham, A., and El Sayed, Z. FaithfulSAE: Towards capturing faithful features with sparse autoencoders without external dataset dependencies, 2025. URL https://arxiv.org/abs/2506.17673.

Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

Fel, T., Lubana, E. S., Prince, J. S., Kowal, M., Boutin, V., Papadimitriou, I., Wang, B., Wattenberg, M., Ba, D. E., and Konkle, T. Archetypal SAE: Adaptive and stable dictionary learning for concept extraction in large vision models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=9v1eW8HgMU.

Gadgil, S., Lin, C., and Lee, S.-I. Ensembling sparse autoencoders, 2025. URL https://arxiv.org/abs/2505.16077.

Gao, L., la Tour, T. D., Tillman, H., Goh, G., Troll, R., Radford, A., Sutskever, I., Leike, J., and Wu, J. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tcsZt9ZNKD.

Glivenko, V. Sulla determinazione empirica delle leggi di probabilità. *Giornale dell'Istituto Italiano degli Attuari*, 4:92–99, 1933.

Heap, T., Lawson, T., Farnik, L., and Aitchison, L. Automated interpretability metrics do not distinguish trained and random transformers, 2025. URL https://arxiv.org/abs/2501.17727.

Karvonen, A., Rager, C., Lin, J., Tigges, C., Bloom, J., Chanin, D., Lau, Y.-T., Farrell, E., McDougall, C., Ayonrinde, K., Wearden, M., Conmy, A., Marks, S., and Nanda, N. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability. *arXiv preprint arXiv: 2503.09532*, 2025. URL https://arxiv.org/abs/2503.09532.

Laptev, D., Balagansky, N., Aksenov, Y., and Gavrilov, D. Analyze feature flow to enhance interpretation and steering in language models, 2025. URL https://arxiv.org/abs/2502.03032.

Leask, P., Bussmann, B., Pearce, M. T., Bloom, J. I., Tigges, C., Al Moubayed, N., Sharkey, L., and Nanda, N. Sparse autoencoders do not find canonical units of analysis. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=9ca9eHNrdH.

Leask, P., Bussmann, B., Pearce, M. T., Bloom, J. I., Tigges, C., Moubayed, N. A., Sharkey, L., and Nanda, N. Sparse autoencoders do not find canonical units of analysis. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025b. URL https://openreview.net/forum?id=9ca9eHNrdH.

Paulo, G. and Belrose, N. Sparse autoencoders trained on the same data learn different features, 2025a. URL https://arxiv.org/abs/2501.16615.

Paulo, G. and Belrose, N. Sparse autoencoders trained on the same data learn different features, 2025b. URL https://arxiv.org/abs/2501.16615.

Rajamanoharan, S., Lieberum, T., Sonnerat, N., Conmy, A., Varma, V., Kramár, J., and Nanda, N. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders, 2024. URL https://arxiv.org/abs/2407.14435.

Wang, J., Ge, X., Shu, W., Tang, Q., Zhou, Y., He, Z., and Qiu, X. Towards universality: Studying mechanistic similarity across language model architectures. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=2J18i8T0oI.

Wang, S. L., Quach, A., Parsan, N., and Yang, J. J. Enforcing orderedness to improve feature consistency, 2025b. URL https://arxiv.org/abs/2512.02194.

A. SAE Training Details

During training, we normalize decoder weights so that feature embeddings and gradients have unit norm. Namely, suppose that w_i and g_i is a decoder column and its gradient correspondingly, then the procedure is:

1. Compute row-normalized decoder weights: $u_i = \frac{w_i}{\|w_i\|_2}$.
2. Remove the component parallel to u_i from the gradient: $g_i \leftarrow g_i - (g_i^T u_i)u_i$, so that gradient step will be performed on the tangent space of the unit sphere.
3. Set weights to normalized: $w_i \leftarrow u_i$.

After training, we fold the mean and the standard deviation of the initial hidden state buffer into the weights so that no explicit normalization of the input and rescaling of the output is required, then normalize the decoder and scale encoder weights and bias to counteract this normalization, so that SAE do not change the output.

B. Additional Details for Quantitative Analysis

B.1. Evaluation tokens and activation collection

All quantitative statistics (frequencies, magnitudes, entropy, EV, and CE under patching) are computed over a held-out collection of token positions. Let B_{eval} be the number of evaluation batches and T the context length, and define

$$N_{\text{tok}} \;=\; B_{\text{eval}} \cdot T.$$

For token index $n \in \{1, \ldots, N_{\text{tok}}\}$, let $\boldsymbol{h}_n \in \mathbb{R}^d$ denote the base-model activation at the SAE training location (residual stream), and let $\boldsymbol{z}_n \in \mathbb{R}^F_{\geq 0}$ denote SAE feature activations. We use the SAE reconstruction

$$\hat{\boldsymbol{h}}_n = \boldsymbol{W}_{\text{dec}} \boldsymbol{z}_n + \boldsymbol{b}_{\text{dec}}. \tag{6}$$

All evaluation batches are disjoint from SAE training batches.

B.2. Cosine-threshold sensitivity across SAE families

We vary the cosine matching threshold θ used to declare cross-seed feature matches and recompute the endpoint fractions for each SAE family in the main GPT-2 setting. Figure 10a shows that increasing θ makes matching stricter, so the stable fraction falls and the unstable fraction rises across all SAE types. The qualitative ordering is preserved throughout the sweep: TopK and BatchTopK remain nearly identical, JumpReLU stays intermediate, and Vanilla exhibits the strongest endpoint stability.

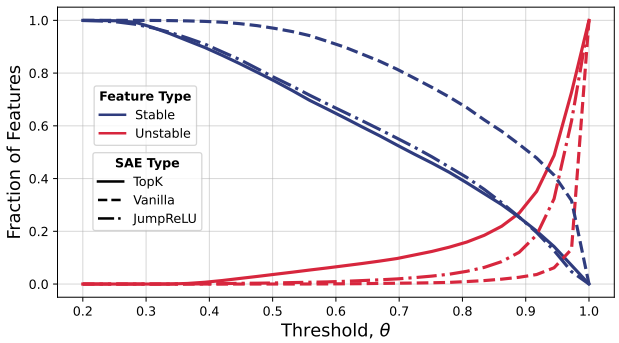
B.3. Usage statistics: activation frequency and mean magnitude

We quantify how often each feature is used and how strongly it fires when active via:

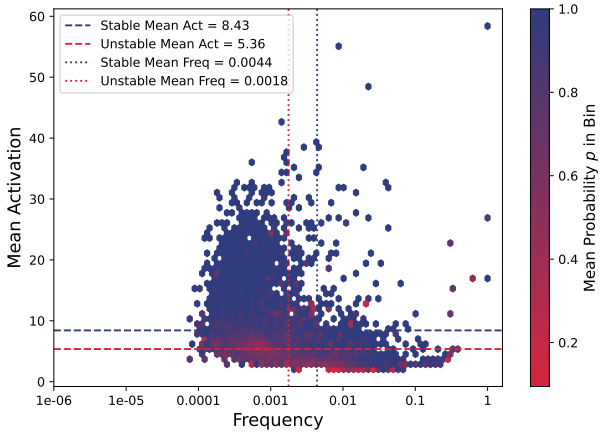
$$\omega_i = \frac{1}{N_{\text{tok}}} \sum_{n=1}^{N_{\text{tok}}} \mathbf{1}\{z_{n,i} > 0\}, \tag{7}$$

$$\mu_i = \frac{\sum_{n=1}^{N_{\text{tok}}} z_{n,i}}{\sum_{n=1}^{N_{\text{tok}}} \mathbf{1}\{z_{n,i} > 0\}}. \tag{8}$$

Here ω_i is the fraction of evaluated token positions where feature i activates, and μ_i is the mean activation magnitude conditioned on activation.



(a) **Endpoint fractions vs. cosine threshold θ.** Fractions of *stable* and *unstable* features as a function of cosine matching threshold θ for several SAE types in the same base-model setting.



(b) **Activation frequency and conditional mean magnitude for stable vs. unstable features.** Unstable features are concentrated at lower frequencies, while stable features exhibit a heavier high-magnitude tail.

Figure 10. **Additional quantitative appendix results.** Left: threshold sensitivity of endpoint fractions across SAE families. Right: joint distribution of activation frequency ω_i and conditional mean magnitude μ_i for stable and unstable features.

B.4. Token entropy computation

Let $v_n \in \{1, \dots, |\mathcal{V}|\}$ be the vocabulary token id at token position n. For each feature i, define token-conditional activation counts and the induced token distribution

$$c_i(v) = \sum_{n=1}^{N_{\text{tok}}} \mathbf{1}\{(v_n = v) \wedge (z_{n,i} > 0)\}, \tag{9}$$

$$\pi_i(v) = \frac{c_i(v)}{\sum_{v' \in \mathcal{V}} c_i(v')}, \tag{10}$$

and token entropy

$$H_i = -\sum_{v \in \mathcal{V}} \pi_i(v) \log \pi_i(v). \tag{11}$$

We use the natural logarithm. Features that never activate on the evaluation set are excluded from entropy plots.

B.5. Masking protocol, uncertainty, and reweighting

For a masked index set $M \subseteq \{1, \dots, F\}$, define

$$\boldsymbol{z}_n^{(-M)} = \boldsymbol{z}_n \odot (\mathbf{1} - \mathbf{1}_M), \tag{12}$$

$$\hat{\boldsymbol{h}}_n^{(-M)} = \boldsymbol{W}_{\text{dec}} \boldsymbol{z}_n^{(-M)} + \boldsymbol{b}_{\text{dec}}. \tag{13}$$

We measure reconstruction quality by explained variance

$$\text{EV}(M) = 1 - \frac{\mathbb{E}_n\left[\|\boldsymbol{h}_n - \hat{\boldsymbol{h}}_n^{(-M)}\|_2^2\right]}{\mathbb{E}_n\left[\|\boldsymbol{h}_n - \bar{\boldsymbol{h}}\|_2^2\right]}, \qquad \bar{\boldsymbol{h}} = \mathbb{E}_n[\boldsymbol{h}_n]. \tag{14}$$

For each masking budget N, we sample masks as follows:

- Sample M_S by drawing N features uniformly without replacement from $\mathcal{S}_\varepsilon$.
- Sample M_U by drawing $4N$ features uniformly without replacement from $\mathcal{U}_\varepsilon$.

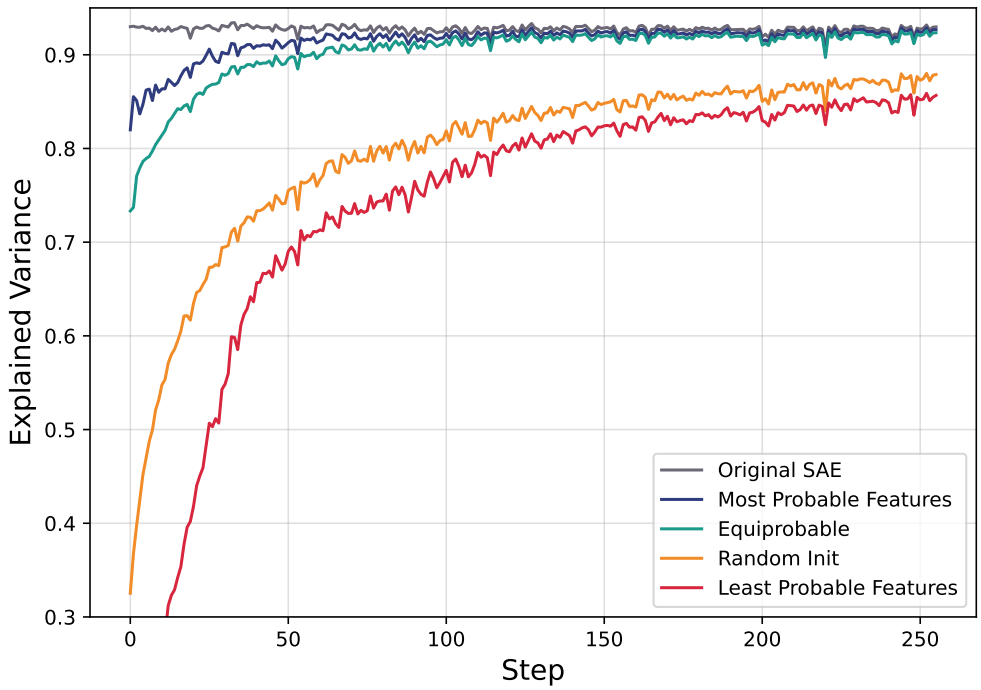


Figure 11. **Explained variance during brief tuning of SAEs constructed from different feature subsets.** Initialization from the most probable features recovers nearly the same explained variance as the original SAE, the equiprobable construction performs somewhat worse, and the least-probable construction lags substantially behind.

We repeat this sampling procedure $R = 10$ times independently for each N. Plots report the mean across these R draws, and the shaded band indicates one standard deviation across draws.

Reweighting. In addition to direct masked reconstructions $\hat{\boldsymbol{h}}_n^{(-M)}$, we evaluate a reweighted variant that matches the ℓ_2 norm of the unmasked SAE reconstruction $\hat{\boldsymbol{h}}_n$ at each token position:

$$\tilde{\boldsymbol{h}}_n^{(-M)} = \hat{\boldsymbol{h}}_n^{(-M)} \cdot \frac{\|\hat{\boldsymbol{h}}_n\|_2}{\|\hat{\boldsymbol{h}}_n^{(-M)}\|_2 + \delta}, \tag{15}$$

where $\delta > 0$ is a small constant for numerical stability.

B.6. Next-token loss under activation patching

To measure downstream impact, we patch the residual stream at the SAE training location. For each token position n, we run a clean forward pass and compute cross-entropy $\mathrm{CE}_{\text{base}}$ on the evaluation tokens. We then rerun the forward pass where $\boldsymbol{h}_n$ is replaced by either $\hat{\boldsymbol{h}}_n^{(-M)}$ (no reweighting) or $\tilde{\boldsymbol{h}}_n^{(-M)}$ (with reweighting), and compute $\mathrm{CE}_{\text{patched}}(M)$ on the same evaluation tokens. We report

$$\Delta\mathrm{CE}(M) = \mathrm{CE}_{\text{base}} - \mathrm{CE}_{\text{patched}}(M), \tag{16}$$

so more negative values indicate larger degradation (as in Figure 3).

B.7. Explained variance during tuning of constructed SAEs

B.8. Probability distributions of selected features in constructed SAEs

C. Additional Results for the Controlled Low-Rank Synthetic Model

We report additional synthetic-model results beyond the main setting from Section 7. Across multiple choices of low-rank dimension r and sparsity level k, the same qualitative pattern persists: full-rank features are consistently stable and well recovered, while features confined to the shared low-rank block are much less stable as individual directions.

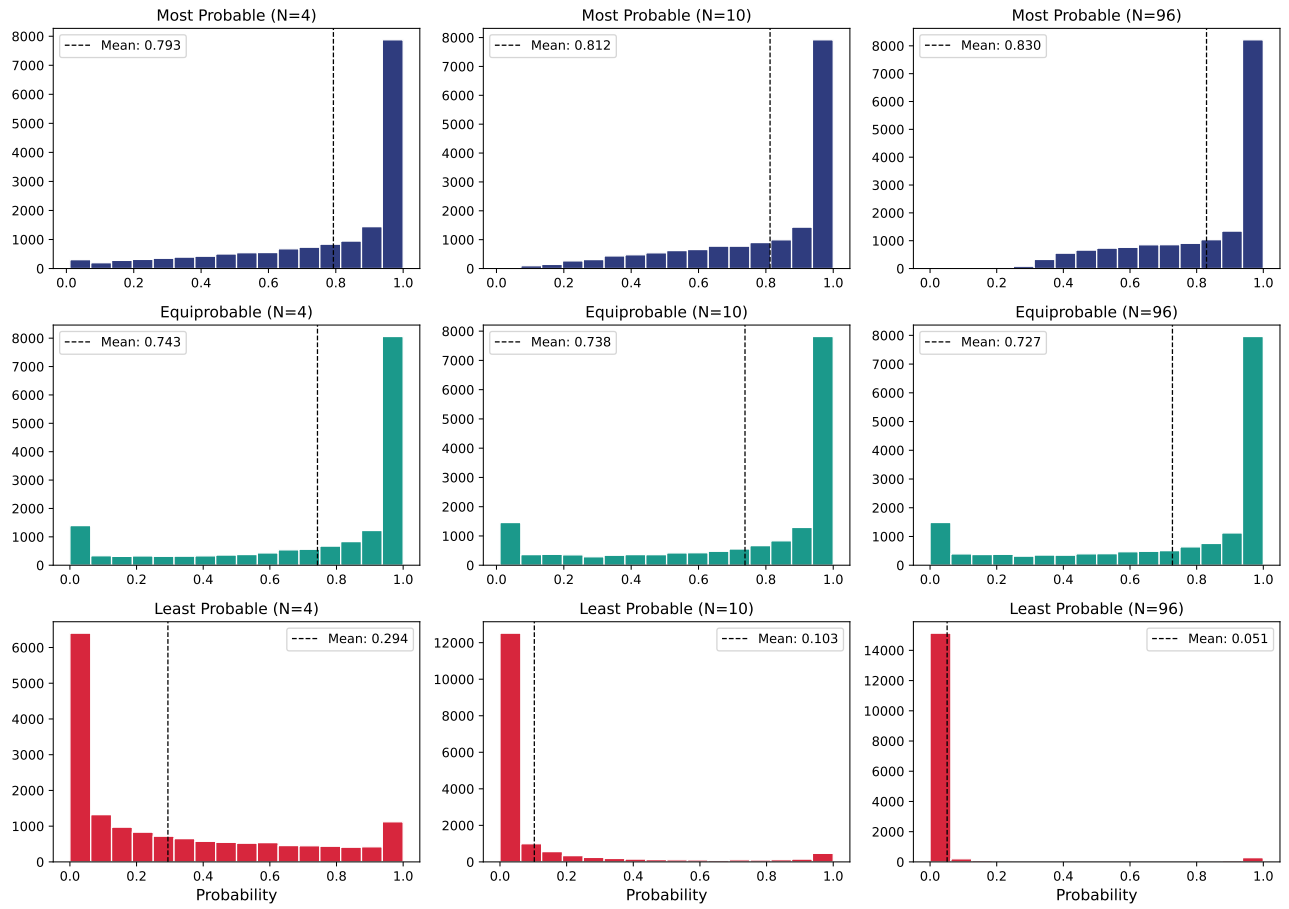


Figure 12. **Probability distributions of selected features for different source-pool sizes.** Rows correspond to the most-probable, equiprobable, and least-probable constructions; columns use $n = 4$, 10, and 96 source SAEs. Even small source pools already make the most-probable construction concentrate on high-reappearance features, while the least-probable construction remains dominated by near-zero-probability features.

C.1. Automatic interpretation for baseline vs. most-probable-feature SAE

D. Additional Details for Qualitative Analysis

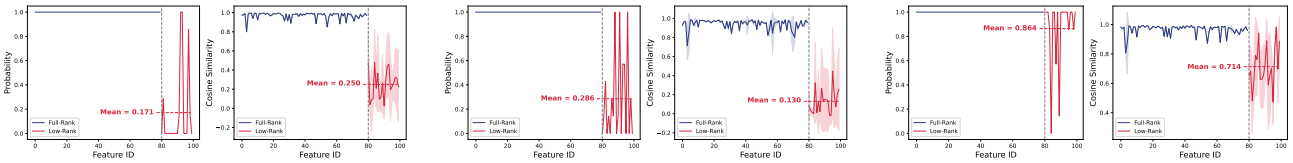
This appendix contains the detailed qualitative results summarized in the token-entropy discussion of Section 5. We compare stable and unstable features using automatic feature interpretation and explanation text analysis.

D.1. Qualitative results from automatic interpretation

We use the same stable/unstable split as elsewhere in the paper (with $\varepsilon = 0.05$ and a single anchor SAE). We run the SAEBench auto-interpretation pipeline (Karvonen et al., 2025) with `Qwen/Qwen3-32B` as the interpreter/evaluator model; full settings are given below in Appendix D.2.

Stable features are more interpretable. SAEBench assigns each feature a *detection score*, defined as the evaluator model's accuracy at predicting whether a given context contains *any* token position on which the feature activates (Appendix D.3). Figure 15b shows the distribution of detection scores for stable and unstable features. Stable features achieve noticeably higher scores overall. This gap is driven primarily by the mass at score $= 1$ (perfect detection on held-out contexts): in our setting, the fraction of stable features with score 1 is $\approx 4.5\times$ larger than for unstable features.

Explanations differ systematically. Inspecting the generated explanations reveals a consistent qualitative split. Unstable features are frequently described in terms of *surface form*: letter patterns, capitalization, prefixes, punctuation, and short



(a) $d = 32, r = 1, k = 4$. *(b)* $d = 32, r = 1, k = 8$. *(c)* $d = 32, r = 4, k = 4$.

Figure 13. **Additional synthetic settings.** The same full-rank versus low-rank split appears across multiple values of the subspace rank r and sparsity level k: full-rank features retain high reappearance probabilities and high cosine similarity to ground truth, whereas low-rank features do not.

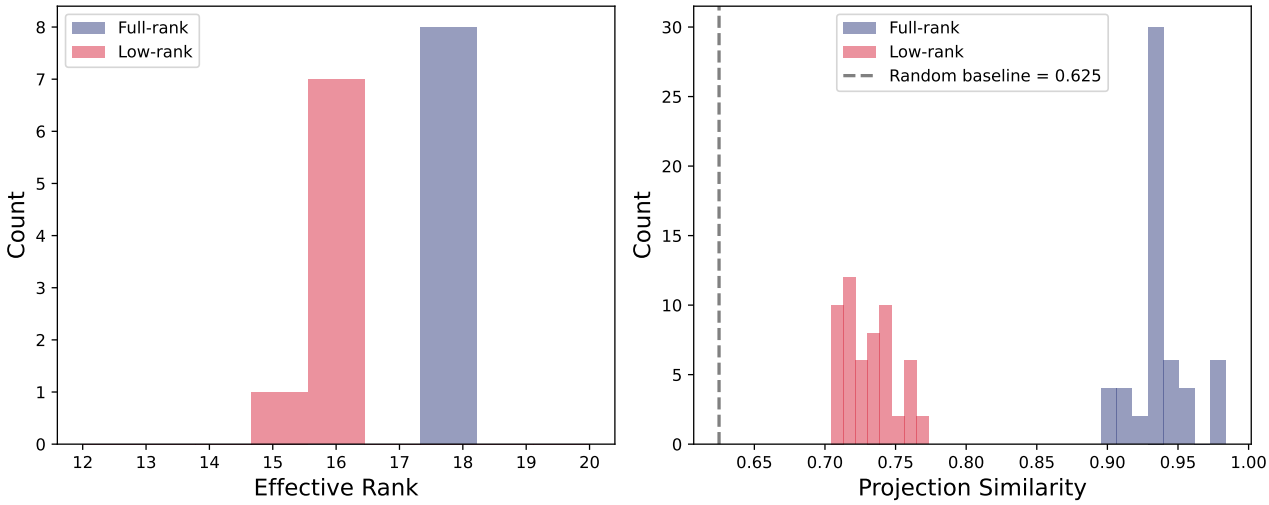


Figure 14. **Effective rank and cross-seed subspace similarity in the synthetic model** ($d = 32$, $r = 2$, $k = 8$)**.** The learned low-rank block has smaller effective rank than the full-rank block, yet its cross-seed subspace similarity remains clearly above the random baseline. Thus the low-rank features are unstable as individual vectors but still reconstruct a reproducible shared subspace.

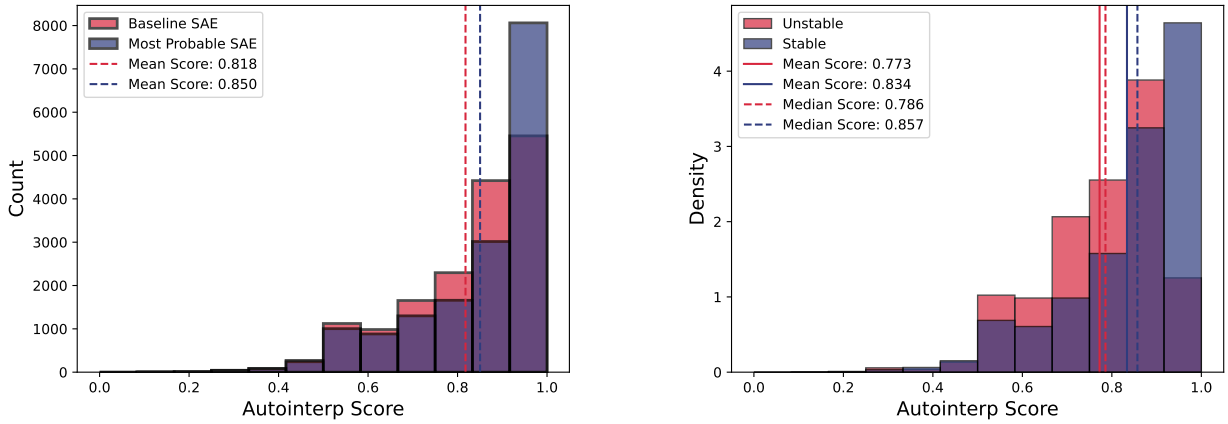
substrings. Stable features more often describe *how words function in context*: phrases and constructions, syntactic roles (e.g., pronouns, conjunctions), or broader semantic groupings. This aligns with Section 5, where unstable features concentrate on lower token-entropy regimes dominated by punctuation/subword triggers, while stable features extend to higher-entropy lexical and conceptual patterns.

Keyword evidence for surface vs. structural patterns. To quantify the surface-vs-structural distinction in explanation text, we measure how often explanations contain keywords such as `substring`, `starting`, `letter` (surface-form) versus `phrase`, `expression`, `construction` (structural/compositional), across bins of $\hat{p}(\boldsymbol{f}_i^{(0)})$. We restrict to explanations with detection score > 0.7 to reduce noise from low-quality explanations. A simple summary already shows a large separation: in the unstable features' explanations, `substring` appears in 38.9% of explanations, while for the stable ones it appears in 11.3%; conversely, `phrase` increases from 4.1% to 32.0% from unstable to stable features. The full table (with confidence intervals and bin counts) is given in Appendix D.5.

Explanations alone predict stability. As an auxiliary test, we ask a frontier LLM to classify features as stable vs. unstable using *only* their auto-interp explanations, with labels anonymized during prompting. On a balanced evaluation set, the model achieves 0.88 accuracy. Full prompts and the exact protocol are given in Appendix D.6.

D.2. Auto-interpretation setup (SAEBench)

We use the SAEBench auto-interpretation pipeline (Karvonen et al., 2025) with `Qwen/Qwen3-32B` as both interpreter and evaluator model. Interpretation/evaluation contexts are drawn from a held-out slice of FineWeb consisting of 2M tokens. For each feature, we build prompt contexts from token windows around activations and cap the model's generation length at 128 tokens.



(a) **Automatic interpretation scores: baseline SAE vs. SAE constructed from the most probable features.** Histogram of detection scores; dashed lines show mean scores for each SAE.

(b) **Automatic interpretation (detection) scores for stable vs. unstable features.** Stable features achieve higher scores, with a substantially larger mass at score $= 1$.

Figure 15. **Automatic-interpretation comparisons.** Left: the most-probable-feature SAE preserves high detection scores relative to the baseline SAE. Right: stable features are substantially easier to detect automatically than unstable ones.

Table 2. **Representative automatic interpretations.** Three illustrative explanations per class are shown for unstable vs. stable features.

Unstable features (examples)	Stable features (examples)
words starting with "ve" followed by a letter that continues a root or morpheme	the phrase "so as to" and similar constructions indicating purpose or intent
proper nouns, especially those starting with capital letters	reflexive pronouns and words referring to oneself or others
the substring "Tor" at the start of words	technical terms related to servers, connections, and computing components

Example selection. For each feature, we construct two disjoint sets of contexts:

- **Explanation set (used to generate the explanation):** 10 contexts with the largest activation value, plus 5 additional contexts sampled via importance sampling.
- **Evaluation set (used to score the explanation):** 2 additional top-activation contexts (not overlapping with the explanation set), plus 10 importance-sampled contexts, plus 2 random contexts.

All sampling is performed independently per feature. The resulting prompts follow SAEBench defaults, aside from the local inference adjustments described above.

D.3. Detection score definition

Given a feature i and a context window c (a sequence of tokens), define the binary label

$$y_i(c) = \mathbf{1}\{\exists \text{ a token position in } c \text{ such that } z_{t,i} > 0\},$$

i.e., whether the context contains *any* position where the feature activates. Given the interpreter-produced explanation text $\mathcal{E}_i$, the evaluator model predicts $\hat{y}_i(c) \in \{0, 1\}$. The *detection score* for feature i is the accuracy on the held-out evaluation set:

$$\mathrm{DS}_i \;=\; \frac{1}{|\mathcal{C}_i|} \sum_{c \in \mathcal{C}_i} \mathbf{1}\{\hat{y}_i(c) = y_i(c)\},$$

where $\mathcal{C}_i$ is the evaluation context set described in Appendix D.2.

Table 3. **Additional automatic interpretations.** More examples of unstable (surface-form) vs. stable (structural/compositional) explanations. Only interpretations with the score > 0.7 are shown.

Unstable features	Stable features
words starting with "Div" or containing the substring "div" in various contexts	*phrases involving positional prepositions like "in the," "right up to," and "next to."*
the substring "wr" at the start of words	*phrases indicating reference to previously mentioned information, such as "as described above" or "mentioned earlier"*
the substring "Tro" at the start of words	*phrases describing characters and their roles in stories or films*
words starting with the substrings "wa", "por", "po", or similar letter combinations	*words related to peace and calmness, including variations like "peaceable" and "peacefully"*
partial words with double letters in the middle, such as "Perr", "Phys", "Ol", "Bonni", "feat", "iodin", and "Coleman"	*the word "firm" and similar capitalized terms in business or organizational contexts*
words containing the substring "cran" followed by a letter, often related to "cranberries" or "cranium."	*words starting with the letter "L" (case-insensitive)*
first names, especially when preceded by a description or title	*words ending in the substring "-el"*
words starting with "Rum", "mist", and "of" as well as variations like "Rumor", "misten", and "of"	*the word "lock" and variations with attached punctuation or nearby words*
words containing the substring "br" followed by "on" or "o" in medical or anatomical contexts	*the phrase "no longer" and related negated continuations*
words starting with "El" or "el"	*names and parts of names starting with "Mc" or "McK" or "Mac"*
surnames and partial surnames, especially those starting with "Hart"	*words related to negative outcomes or effects, especially when paired with terms like "impact," "effects," or "feedback."*
the word "batch" and similar capitalized substrings like "Dow" and "Alabama"	*words indicating uncertainty or hypothetical situations, such as "potential," "possible," and similar variations*
names or parts of names with a specific stylistic or phonetic pattern, particularly those beginning with `<<Alfred>>`, `<<styl>>`, `<<transc>>`, *or* `<<University>>`	*phrases involving change with quantity or degree, especially those using "the… the…" or "increases/decreases with."*
words related to storage containers, especially "tank," "tanks," and variations like "gallons," "empty," and "tankers."	*the word "healthy" and related concepts about diet and well-being*
words starting with "pie" or "piez" and related technical or compound terms	*variations of the word "dig" and related actions or contexts*
the surname "Gordon" and related proper nouns	*adjectives and phrases starting with "un-" that convey a sense of freedom or lack of restriction*
words starting with the substring "Whit" or "Wal"	*coordinating conjunctions "and" and "or" and their surrounding context*
government-related terms such as "Cabinet," "Prime," and "bills."	*words related to teenagers and specific substrings like "teen", "teens", "Teen", and "agers"*
words related to substances, fluids, or materials in contexts of waste, water, or bodily fluids	*the word "course" and related educational concepts*
capitalized names and the substring "Marshall" with variations	*language names and related terms such as English, Spanish, Arabic, and their variations*

D.4. Additional representative interpretations

Table 2 above shows a small set of representative explanations illustrating the surface-form vs. structural distinction. Here we provide additional examples to support the qualitative claims in Appendix D.

D.5. Keyword analysis across stability bins

We bin features by their empirical reappearance rate $\hat{p}(\boldsymbol{f}_i^{(0)})$ and compute, for each bin and each keyword k, the empirical fraction $\hat{q}$ of explanations that contain k as a substring match (case-sensitive, using the raw explanation text). To reduce noise from low-quality explanations, we restrict to features with detection score $\mathrm{DS}_i > 0.7$.

We report confidence intervals for each proportion. Below we include two versions of the same table: a Wald interval (symmetric; matches the common $\hat{q} \pm 1.96\sqrt{\hat{q}(1-\hat{q})/n}$ form) and a Wilson interval (typically better calibrated near 0/1).

D.6. LLM-only classification from explanation text

We test whether the *auto-interp explanation text alone* contains enough signal to predict feature stability. Concretely, we run a blind classification experiment where a frontier LLM (GPT-5) receives only short natural-language explanations produced by the auto-interpretation pipeline summarized in Appendix D and must predict a binary label. No activation traces, token examples, decoder vectors, or $\hat{p}$ values are shown.

Data and split. We construct a balanced dataset of explanations from endpoint features with $\varepsilon = 0.05$ using the same anchor SAE as in the main text. We sample 50 explanations from each class for in-context training (100 total) and a disjoint 50 explanations from each class for evaluation (100 total). During prompting, labels are anonymized as $\{0, 1\}$ and the model

Table 4. **Keyword frequencies in feature explanations across reappearance bins (Wald 95% CI).** Each entry reports the fraction of explanations containing the given keyword, shown as $\hat{q}\pm$ (95% CI). Column headers give bin ranges for $\hat{p}(\boldsymbol{f}_i^{(0)})$ and the number of features n in that bin after restricting to detection score > 0.7.

Keyword	$[0.00, 0.05)$ (n=1020)	$[0.05, 0.10)$ (n=220)	$[0.90, 0.95)$ (n=1053)	$[0.95, 1.00]$ (n=6547)
substring	0.389±0.030	0.209±0.054	0.121±0.020	0.113±0.008
starting	0.276±0.027	0.082±0.036	0.050±0.013	0.052±0.005
letter	0.108±0.019	0.068±0.033	0.036±0.011	0.037±0.005
phrase	0.041±0.012	0.205±0.053	0.313±0.028	0.320±0.011
expression	0.010±0.006	0.036±0.025	0.056±0.014	0.058±0.006
construction	0.008±0.005	0.032±0.023	0.066±0.015	0.060±0.006

is not told what the labels correspond to.

Classification prompt. We use a single in-context learning prompt: first provide labeled training examples (balanced across labels), then ask for labels on the test set, requesting one label per line and no additional text. We report accuracy on the balanced test set.

```
You will see short explanations of patterns that activate a feature.
Each explanation has a label 0 or 1. Learn what distinguishes the labels.

TRAINING EXAMPLES:
Label: 0  Explanation: <EXPLANATION_1>
Label: 0  Explanation: <EXPLANATION_2>
...
Label: 1  Explanation: <EXPLANATION_51>
Label: 1  Explanation: <EXPLANATION_52>
...

Now predict labels for the TEST explanations.
Output one label (0 or 1) per line, in order, with no extra text.

TEST EXPLANATIONS:
1) <TEST_EXPLANATION_1>
2) <TEST_EXPLANATION_2>
...
```

Result. On the balanced test set (50 explanations per class), GPT-5 achieves accuracy 0.88. Since the model sees only explanation text, this indicates that stability is strongly reflected in the linguistic descriptions produced by auto-interpretation.

Qualitative class summaries prompt. In addition to classification, we ask the same LLM to summarize typical patterns in each class, using explanations only. We provide a small random subset from each class (without mentioning stability) and request concise descriptions.

```
Here are explanations from two classes (A and B). Summarize typical patterns in each class.
Be concise and concrete.

Class A:
- <EXPLANATION_A1>
- <EXPLANATION_A2>
...

Class B:
- <EXPLANATION_B1>
- <EXPLANATION_B2>
```

Table 5. Effective rank for size-matched probability sets.

Probability Set	Effective Rank
$p \leq 0.01$	450.5 ± 4.4
$p \leq 0.05$	498.5 ± 4.4
$p \geq 0.95$	620.8 ± 3.2
$p \geq 0.99$	617.7 ± 3.3

...

Model-produced summaries. The model consistently describes the two classes in terms of surface-form versus structural/compositional language. In a representative run it summarized:

- **Unstable:** explanations emphasize what the word looks like on the surface (substrings, prefixes, capitalization, punctuation) and narrow lexical triggers.
- **Stable:** explanations emphasize how words function in sentences (phrases, constructions, discourse roles) and broader compositional or semantic groupings.

Notes and limitations. This experiment is intended as a lightweight diagnostic rather than a primary result. Because it relies on a particular evaluator model and prompt format, we treat it as supportive evidence that the surface-vs-structural distinction is present in explanation text, consistent with Table 2 and the keyword analysis in Appendix D.5.

E. Effective Rank and SVD of the Decoder Submatrices

As in the main text, given an index set $\mathcal{I}_\varepsilon^{(s)} \subseteq \{1, \ldots, F\}$ (either $\mathcal{U}_\varepsilon^{(s)}$ for unstable features or $\mathcal{S}_\varepsilon^{(s)}$ for stable features), define the corresponding submatrix

$$X_{\mathcal{I}_\varepsilon}^{(s)} \;:=\; \boldsymbol{W}_{\text{dec}}^{(s)}[:, \mathcal{I}_\varepsilon^{(s)}] \in \mathbb{R}^{d \times m_s}, \qquad m_s := |\mathcal{I}_\varepsilon^{(s)}|.$$

We consider its singular value decomposition

$$X_{\mathcal{I}_\varepsilon}^{(s)} \;=\; V_\varepsilon^{(s)}\, \Sigma_\varepsilon^{(s)} \left(U_\varepsilon^{(s)}\right)^\top,$$

where $V_\varepsilon^{(s)} \in \mathbb{R}^{d \times d}$ and $U_\varepsilon^{(s)} \in \mathbb{R}^{m_s \times m_s}$ are orthogonal matrices, and $\Sigma_\varepsilon^{(s)} \in \mathbb{R}^{d \times m_s}$ is rectangular, with diagonal entries $\sigma_1^{(s)} \geq \sigma_2^{(s)} \geq \cdots \geq \sigma_d^{(s)} \geq 0$, that are singular values, and all remaining diagonal entries are zero.

We define the effective rank of submatrix $X_{\mathcal{I}_\varepsilon}^{(s)}$ as

$$\mathrm{ER}(X_{\mathcal{I}_\varepsilon}^{(s)}) = \exp\Big(- \sum_{k=1}^{\min(d, m_s)} p_k^{(s)} \, \log p_k^{(s)} \Big),$$

where $p_k^{(s)} = \sigma_k^{(s)} / \sum_{j=1}^{\min(d,m_s)} \sigma_j^{(s)}$. To fairly compare the effective ranks of submatrices with different numbers of features, we calculate them under a size-matched protocol: for each seed for all feature sets we select

$$k^{(s)} \;=\; \min\left(|\mathcal{U}_{0.01}^{(s)}|,\; |\mathcal{S}_{0.01}^{(s)}|, |\mathcal{U}_{0.05}^{(s)}|,\; |\mathcal{S}_{0.05}^{(s)}|\right),$$

and for sets whose number of features differs from $k^{(s)}$, we estimate ER by averaging over $B = 50$ random subsamples of size $k^{(s)}$.

We define the explained variance $\mathrm{EV}_{\text{SVD}}^{(s)}(r)$ of singular values of matrix $X_{\mathcal{I}_\varepsilon}^{(s)}$ as

$$\mathrm{EV}_{\text{SVD}}^{(s)}(r) = \sum_{i=1}^{r} \left(\sigma_i^{(s)}\right)^2 / \sum_{i=1}^{d} \left(\sigma_i^{(s)}\right)^2.$$

Table 6. Fractions of *unstable* ($\hat{p} \leq \varepsilon$) and *stable* ($\hat{p} \geq 1 - \varepsilon$) features for TopK SAEs across dictionary sizes and layers.

Base model	Layer	$F = 2^{14}$		$F = 2^{15}$		$F = 2^{16}$	
		Unstable	Stable	Unstable	Stable	Unstable	Stable
GPT-2	4	0.219	0.339	0.161	0.356	0.088	0.387
	7	0.080	0.447	0.066	0.440	0.051	0.432
	10	0.031	0.484	0.035	0.470	0.040	0.433
Pythia-160M-deduped	4	0.136	0.421	0.109	0.558	0.049	0.634
	7	0.062	0.513	0.096	0.569	0.468	0.316
	10	0.064	0.461	0.077	0.527	0.114	0.456
Gemma-2 2B	7	0.042	0.653	0.027	0.647	0.021	0.588
	12	0.013	0.563	0.012	0.548	0.024	0.482
	18	0.005	0.681	0.008	0.629	0.020	0.509

Let $V_\varepsilon^{(s)}[:, 1{:}r] \in \mathbb{R}^{d \times r}$ denotes the matrix, whose r columns are the top-r left singular vectors, and define the projector onto their span

$$P_\varepsilon^{(s)}(r) \;:=\; V_\varepsilon^{(s)}[:, 1{:}r]\,(V_\varepsilon^{(s)}[:, 1{:}r])^\top.$$

Then $\mathrm{EV}_{\mathrm{SVD}}^{(s)}(r)$ admits an alternative representation

$$\mathrm{EV}_{\mathrm{SVD}}^{(s)}(r) \;=\; \frac{\left\|P_\varepsilon^{(s)}(r) X_{\mathcal{I}_\varepsilon}^{(s)}\right\|_F^2}{\left\|X_{\mathcal{I}_\varepsilon}^{(s)}\right\|_F^2}.$$

This suggests a natural cross-seed analogue: for an anchor seed a and another seed $s \neq a$, define

$$\mathrm{EV}_{\mathrm{SVD}}^{(a\to s)}(r) \;:=\; \frac{\left\|P_\varepsilon^{(a)}(r) X_{\mathcal{I}_\varepsilon}^{(s)}\right\|_F^2}{\left\|X_{\mathcal{I}_\varepsilon}^{(s)}\right\|_F^2}.$$

Intuitively, $\mathrm{EV}_{\mathrm{SVD}}^{(a\to s)}(r)$ measures how well the top-r singular subspace learned in seed a explains the feature subspace in seed s.

F. Additional Details for Other Setups

F.1. Endpoint fractions across models, layers, and dictionary sizes

Section 8 summarizes how endpoint stability varies across base models, layers, and dictionary sizes for TopK SAEs. Table 6 reports the full endpoint fractions for all configurations considered.

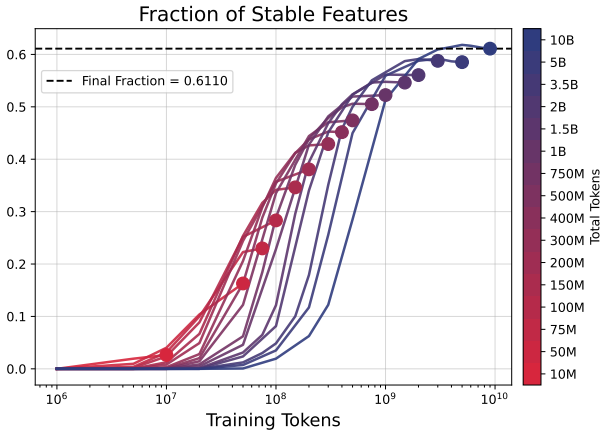
F.2. Training-token sweeps: setup and measurement details

In Section 8 we study how endpoint stability evolves as SAE training progresses. We sweep the total number of SAE training tokens from 10M to 10B while keeping the SAE architecture and hyperparameters fixed to the main TopK setting (GPT-2, `resid_post` layer 7, TopK=64, $F = 2^{14}$). For each training budget, we train the same number of independent SAEs (different random seeds) and estimate feature reappearance probabilities using the matching procedure in Section 4. We then report the endpoint fractions (unstable and stable) using the same ε and cosine threshold θ as in the main setting.

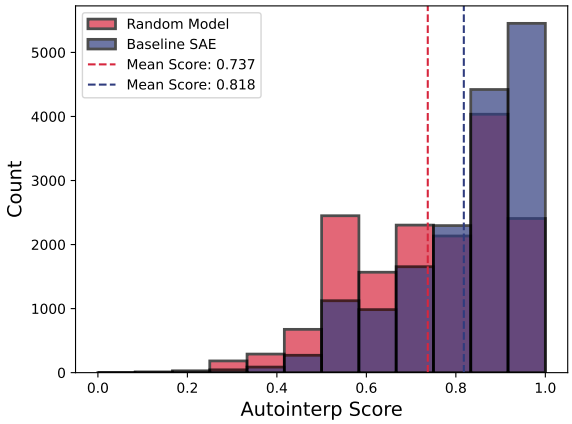
Figure 8 reports the fraction of unstable features as a function of training tokens. Figure 16a reports the corresponding stable fraction curve. Together, these curves show that instability decreases early and then approaches a non-zero plateau, while the stable fraction continues to increase with diminishing returns.

F.3. Automatic interpretation on trained vs. random transformers

Figure 16b compares automatic interpretation (detection) scores for SAEs trained on activations from a trained GPT-2 versus a randomly initialized GPT-2 (same SAE architecture and hyperparameters). This shows that high detection scores can occur even in the random-model setting, motivating stability as a complementary faithfulness signal.



(a) **Stable fraction vs. SAE training tokens.** Fraction of stable features in the main TopK setting as a function of total SAE training tokens.



(b) **Automatic interpretation scores for SAEs trained on trained vs. random base models.** High detection scores are achievable even on random-model activations, despite the absence of cross-seed reproducibility.

Figure 16. **Additional controls and training dynamics.** Left: the stable fraction increases with more SAE training, but with diminishing returns. Right: high automatic-interpretation scores remain possible even for SAEs trained on random-model activations.