PRESERVING LOCAL ACTIVATION STRUCTURE FOR MORE RELIABLE SPARSE AUTOENCODER FEATURES

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse autoencoders (SAEs) provide a way to interpret neural network representations in terms of human-understandable concepts. However, recent evidence of feature instability across training runs and sensitivity to input perturbations raises concerns about the reliability of these interpretations. We argue that preserving the local structure of model activations in their sparse representations allows SAE features to reflect similarities and differences captured by the model. We introduce kNN SAE, which adds a regularizer based on k-nearest neighbors among model activations to incorporate this structural constraint into training. This objective is designed to enable consistent descriptions across related inputs and training runs, making them easier to reproduce and compare. In our experiments, kNN SAE improves feature reproducibility and performance on concept detection and disentanglement evaluations at matched sparsity, with a modest cost in reconstruction and functional fidelity. These findings suggest that local geometric consistency provides a useful foundation for learning more stable and reliable concept representations.

1 INTRODUCTION

Sparse autoencoders (SAEs) represent language model activations as sparse combinations of features that can be inspected and used for intervention (Bricken et al., 2023; Cunningham et al., 2023; Gao et al., 2024). Their usefulness depends on whether these features provide a reliable description of the model's representations. However, learned features can vary across training runs and respond strongly to small input perturbations (Paulo & Belrose, 2025; Gerasimov et al., 2026; Li et al., 2026).

Standard SAE training balances reconstruction quality and sparsity, but does not explicitly constrain how relationships between inputs are represented in latent space. Different sparse decompositions can achieve similar reconstruction quality while organizing their features differently. Accurate reconstruction therefore leaves open questions about the consistency and practical usefulness of the resulting representation.

We investigate whether preserving the local structure of language model representations can improve the reliability of SAE features. This structure provides an additional constraint on how the encoder organizes information into sparse features. We hypothesize that incorporating it into training promotes more reproducible feature decompositions across independent runs and more stable feature selection under adversarial input perturbations.

We introduce kNN SAE, which augments reconstruction with a local distance-preservation objective adapted from (Chen et al., 2022). The method constructs a neighborhood graph over model activations and matches distances along its edges to distances between sparse codes, allowing for a shared scale factor. This constraint operates after sparsification and requires no task labels or alignment between independently trained dictionaries.

Our experiments show that preserving local activation geometry offers a practical route to more reliable SAE features. At matched sparsity, kNN SAE improves neighborhood preservation across three language models, with gains in concept detection, targeted ablation, and resistance to adversarial feature disruption on GPT-2 and Gemma-2-2B. These benefits extend across training runs: on GPT-2 layer 7, regularization reduces the number of features in the lowest-reappearance bin, while expanding the population of consistently recovered features. Selected steering case studies

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

further demonstrate improved concept–fluency trade-offs, connecting representation quality to more controlled interventions. The gains depend on regularization strength and involve a trade-off with reconstruction fidelity. Together, these findings establish local geometry preservation as a useful training principle for learning sparse features that are more reproducible, robust, and effective in downstream applications.

2 PRELIMINARIES

Sparse autoencoders consists of two linear layers with biases with non-linearity in the middle. It inputs hidden state $\mathbf{h} \in \mathbb{R}^d$ and outputs reconstruction $\hat{\mathbf{h}} \in \mathbb{R}^d$. More formally:

$$\mathbf{z} = \sigma(W_{\text{enc}}\mathbf{h} + \mathbf{b}_{\text{enc}})$$
$$\hat{\mathbf{h}} = W_{\text{dec}}\mathbf{z} + \mathbf{b}_{\text{dec}},$$

where $W_{enc} \in \mathbb{R}^{F\times d}$, $\mathbf{b}_{enc} \in \mathbb{R}^F$, $W_{dec} \in \mathbb{R}^{d\times F}$, $\mathbf{b}_{enc} \in \mathbb{R}^d$ are trainable parameters. F is larger than d and called *dictionary size*, $\mathbf{z} \in \mathbb{R}^F$ is *latent activations*. This model is trained with the following loss:

$$\mathcal{L}_{\text{SAE}} = \|\mathbf{h} - \hat{\mathbf{h}}\|_2^2 + \mathcal{L}_{\text{aux}}(\mathbf{f}),$$

where $\mathcal{L}_{\text{aux}}$ usually forces sparsity on $\mathbf{f}$, in simplest case σ is ReLU and $\mathcal{L}_{\text{aux}}(\mathbf{f})$ is $\lambda_{\text{aux}} \cdot \|\mathbf{f}\|_1$.

After the training latent activations $\{z_i(\mathbf{h})\}_{i=1}^F$ used for analysis of hidden state $\mathbf{h}$. However, sparse coding introduces new challenges due to information loss. For example, SAE's latents $\mathbf{z}$ performs worse on sparse probing tasks in comparison to original $\mathbf{z}$, which limits SAE applications for interpretable classifiers. Other limitation is sensitivity to small perturbations of the context (Li et al., 2026). In our work we argue that using additional regularization on latent space $\mathcal{Z}$ is crucial for improving SAE applicability.

3 RELATED WORK

Geometry-preserving autoencoders. A broader line of work has shown that accurate reconstruction does not by itself guarantee preservation of the structure of the input representation. (Lee et al., 2021) introduced Neighborhood Reconstructing Autoencoders, which incorporate local neighborhoods into reconstruction in order to better preserve local connectivity and geometry. (Moor et al., 2020) proposed Topological Autoencoders, which explicitly regularize multiscale connectivity using persistent homology. (Duque et al., 2023) introduced Geometry Regularized Autoencoders, which guide latent representations toward geometry obtained from manifold learning. More recently, (Lim et al., 2024) proposed Graph Geometry-Preserving Autoencoders, where a similarity graph defines the geometry whose distortion is controlled in latent space. Together, these works distinguish pointwise reconstructability from preservation of relations between examples and make the choice of geometry an explicit part of representation learning.

Our closest methodological antecedent is (Chen et al., 2022), who use continuous k-nearest-neighbor (CkNN) graphs to identify local pairs and introduce a scale-adjusted objective that matches distances between neighboring examples before and after encoding. We adopt this principle for sparse representations of LLM activations. In particular, our loss matches nonzero source distances rather than merely attracting neighboring codes, and we study whether imposing this local relational constraint improves the reproducibility of SAE features.

Graph-regularized sparse autoencoders. Graph structure has also been incorporated directly into SAE training. (Nerrise et al., 2026) proposed GeoSAE, which constructs a kNN graph over embeddings from a brain-MRI foundation model and applies graph-Laplacian smoothing to encoder preactivations before the sparsity operation. This encourages neighboring inputs to have similar preactivations, whereas our objective acts on sparse codes after sparsification and preserves the magnitude of local differences rather than only encouraging smoothness. (Yeon et al., 2025) proposed GSAE for LLM safety steering, regularizing decoder directions over a neuron co-activation

graph. Its graph vertices are neurons, whereas the vertices in our graph are individual activation examples. Thus, these approaches use graphs at different levels of the SAE and regularize different mathematical objects.

Feature instability and robustness. (Paulo & Belrose, 2025) showed that changing the initialization seed can substantially change the features recovered from the same model and data. Importantly, (Gerasimov et al., 2026) found that individually unstable features can concentrate in reproducible lower-rank subspaces; their controlled synthetic setting further demonstrates recovery of shared subspaces without identifiable individual latents. This separates reproducibility of representational structure from reproducibility of individual feature axes: preserving geometry alone does not imply axis identifiability. Separately, (Li et al., 2026) showed that small adversarial input perturbations can manipulate SAE concept representations while leaving the underlying LLM activations comparatively unchanged.

4 METHOD

Consider a batch of hidden states $\{\mathbf{h}_i\}_{i=1}^{B}$. We build a neighborhood-based graph $G_b = (V_b, E_b)$, where $V_b = \{1, 2, ..., B\}$, following Chen et al. (2022) we choose continuous kNN graph, where edges defined by the following condition:

$$(i,j) \in E_b \iff \|\mathbf{h}_i - \mathbf{h}_j\|_2 \leq \sqrt{\|\mathbf{h}_i - \mathbf{h}_i^{(k)}\|_2 \cdot \|\mathbf{h}_j - \mathbf{h}_j^{(k)}\|_2},$$

where $\mathbf{h}_i^{(k)}$ is the k-th nearest neighbor of $\mathbf{h}_i$ in a given batch.

In order to enforce geometry preserving of LLM activations in SAE latent space, we adopt the distance preservation auxiliary loss from (Chen et al., 2022), defined as follows:

$$\mathcal{L}_{\text{CkNN}} = \frac{1}{|E_b|} \sum_{(i,j)\in E_b} \left(s_b^{-1} \|\mathbf{h}_i - \mathbf{h}_j\|_2 - \gamma \|\mathbf{z}_i - \mathbf{z}_j\|_2 \right)^2.$$

Here, s_b is the square root of mean-squared distance between vertices of G_b. Notably, the optimal value of γ can be obtained by OLS (see Appendix A.1 for derivation):

$$\gamma^* = \frac{\sum_{(i,j)\in E_b} \|\mathbf{h}_i - \mathbf{h}_j\|_2 \cdot \|\mathbf{z}_i - \mathbf{z}_j\|_2}{s_b \sum_{(i,j)\in E_b} \|\mathbf{z}_i - \mathbf{z}_j\|_2^2}.$$

The final loss for kNN SAE is $\mathcal{L} = \mathcal{L}_{\text{SAE}} + \lambda \cdot \mathcal{L}_{\text{CkNN}}$.

5 EXPERIMENTS

In our main experimental setting, we train TopK SAEs (Gao et al., 2024) and our kNN SAE on GPT-2 (Radford et al., 2019) small, Gemma-2-2B (Gemma Team, 2024) and Llama-3.2-1B (Meta, 2024) on 1B tokens from `sample-10BT` subset of FineWeb. On a single NVIDIA H100 80GB, kNN regularization increased amortized training time by 14.81% for GPT-2 small and 6.58% for Gemma-2-2B in the benchmarked settings; full training and efficiency details are provided in Appendix A. First, we quantitatively show the improvements of neighborhood preservation in SAE latent space and following better performance on downstream tasks from SAEBench. Then, we demonstrate that our kNN SAE improves feature stability across different initialization seeds and its features are more stable under prompt perturbations. Finally, we evaluate steering of SAE features corresponding to different concepts and observe that kNN SAE's features cause less damage to LLM's generation fluency under the same concept scores.

5.1 PRESERVATION OF NEIGHBORHOODS

We first examine how an additional neighborhood-preservation loss affects SAE reconstruction quality. To keep sparsity fixed across these experiments, we use SAEs with TopK activation

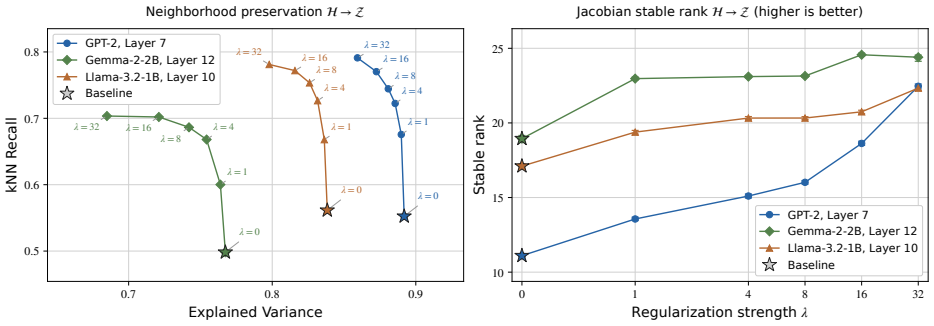

Figure 1: **Neighborhood preservation and encoder geometry.** Effect of regularization on TopK SAEs with $K = 64$ and $F = 16{,}384$ features across all models. Left: mean kNN Recall@16 versus explained variance. Right: median encoder Jacobian stable rank with 95% document-level bootstrap confidence intervals. Stars indicate unregularized baselines ($\lambda = 0$).

($K = 64$) on three models: GPT-2 (Radford et al., 2019) (layer 7), Gemma-2-2B (Gemma Team, 2024) (layer 12), and Llama-3.2-1B (Meta, 2024) (layer 10). We sweep regularization strengths $\lambda \in \{0, 1, 4, 8, 16, 32\}$ and measure explained variance and kNN recall.

For each model, we construct a validation bank of $100{,}000$ token activations from FineWeb documents held out from SAE training. We sample $2{,}048$ query activations per model and reuse them across regularization strengths. We measure neighborhood preservation by mean kNN Recall@16, defined as the fraction of shared neighbors between activation space $\mathcal{H}$ and latent space $\mathcal{Z}$, using Euclidean distances over the full bank and excluding each query itself. For each query, we also compute the stable rank $r_{\text{stable}}(J) = \|J\|_F^2 / \|J\|_2^2$ of the encoder's active Jacobian block with 64 output coordinates, and report the median with 95% document-level bootstrap confidence intervals. Results are shown in Figure 1.

Across all three models, stronger regularization improves neighborhood preservation at the cost of reconstruction quality, with moderate regularization providing a favorable trade-off, suggesting that $\lambda \in \{1, 4\}$ provides a favorable trade-off in these experiments. The encoder Jacobian's stable rank also generally increases, indicating less concentrated sensitivity across singular directions.

To test whether improved neighborhood preservation corresponds to a more faithful representation of interpretable structure, we construct a controlled numerical-role benchmark from real FineWeb text. Each example contains a number used in one of six roles: a currency amount, clock time, calendar date, sports score, citation field, or product specification. We replace the original values with the same approximately uniform set of integers across all roles, preventing either SAE from solving the task through numerical identity alone. We then compare the geometry of Gemma-2-2B (Gemma Team, 2024) layer-18 activations, vanilla TopK SAE codes, and kNN SAE codes with $\lambda = 4$. Full dataset and evaluation details are provided in Appendix B.

Figure 2 shows that the numerical contexts form a branched configuration in the original activation space $\mathcal{H}$. The vanilla TopK SAE largely contracts this structure toward a single diagonal in $\mathcal{Z}$, whereas the kNN SAE partially recovers the distinct branches and transverse variation. This visual difference is also reflected in the full-dimensional metrics. Relative to the vanilla TopK SAE, kNN regularization increases Recall@16 from 0.743 to 0.826, corresponding to approximately 1.34 additional preserved neighbors out of sixteen. The accuracy of a 5-nearest-neighbor role classifier increases from 0.947 to 0.958. For comparison, the same classifier reaches 0.961 directly in $\mathcal{H}$: the vanilla SAE is 1.3 percentage points below the hidden-state reference, while the kNN SAE reduces this gap to 0.3 points. The Spearman correlation between pairwise distances in the two-dimensional PCA projections also increases from 0.720 to 0.773.

These results show that the higher neighborhood recall of the kNN SAE is associated with preservation of meaningful contextual structure, rather than merely improved agreement on an abstract graph statistic. Reconstruction alone retains enough information to keep the six roles broadly separable, but it substantially changes their relative arrangement and local neighborhoods. Explicit

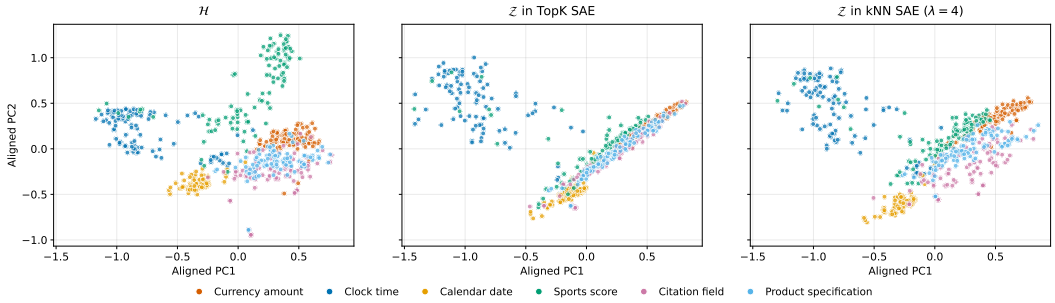

Figure 2: **Geometry of structured numerical contexts.** PCA projections of Gemma-2-2B (Gemma Team, 2024) layer-18 activations $\mathcal{H}$, vanilla TopK SAE codes, and kNN SAE codes with $\lambda = 4$. Colors denote six contextual roles of a numerical token. The vanilla SAE contracts most of the branched structure in $\mathcal{H}$ toward a common diagonal, whereas the kNN SAE preserves more of the original organization. PCA is used only for visualization; the reported 5-NN accuracy and Recall@16 are computed in the full-dimensional spaces.

geometric regularization preserves both more of the original neighbors and more of the higher-level organization visible in $\mathcal{H}$.

5.2 Performance on Downstream Tasks

Having examined neighborhood preservation, we next ask whether kNN regularization improves the usefulness of SAE features for identifying concepts and selectively modifying their contribution to predictions. We evaluate TopK SAEs on GPT-2 (Radford et al., 2019) and Gemma-2-2B (Gemma Team, 2024) at matched sparsity using three SAEBench metrics (Karvonen et al., 2025).

Sparse Probing Top-1 measures how accurately a labeled concept can be predicted from a single selected SAE feature. Spurious Correlation Removal (SCR) evaluates whether ablating features associated with a spurious cue improves a biased classifier's performance on the intended task. Targeted Probe Perturbation (TPP) measures whether ablating features selected for one class preferentially reduces that class's probe accuracy while preserving predictions for other classes. We use an ablation budget of twenty features for SCR and TPP.

Figure 3 shows that kNN regularization improves single-feature probing in both models, making task-relevant information more accessible through individual latents. The gap to probes trained on the original hidden states narrows, although it remains substantial. Several regularized settings also improve both intervention scores relative to the baseline. The response nevertheless differs across models and tasks: GPT-2 (Radford et al., 2019) exhibits sustained gains in probing and TPP, while SCR varies non-monotonically; Gemma-2-2B (Gemma Team, 2024) reaches its strongest results at intermediate coefficients and declines across all three metrics under the strongest regularization.

These improvements occur without using task labels in SAE training: the additional constraint concerns only distances between neighboring activations. This suggests that preserving local relationships can help organize task-relevant information into features that are more useful for classification and targeted ablation. The different responses across metrics also show that regularization strength should be chosen with the intended application in mind.

5.3 Features' Stability

We next examine whether kNN regularization improves feature reproducibility across independently trained SAEs. We evaluate TopK SAEs on GPT-2 (Radford et al., 2019) layers 7 and 10 and Gemma-2-2B (Gemma Team, 2024) layer 7, varying batch-local kNN regularization at fixed architecture and sparsity. Following Gerasimov et al. (2026), we estimate the reappearance probability of feature i in an anchor run a as

$$\hat{p}_i^{(a)} = \frac{1}{S-1} \sum_{s \neq a} \mathbf{1}\left[\max_j \left\langle e_i^{(a)}, e_j^{(s)} \right\rangle \geq \theta \right],$$

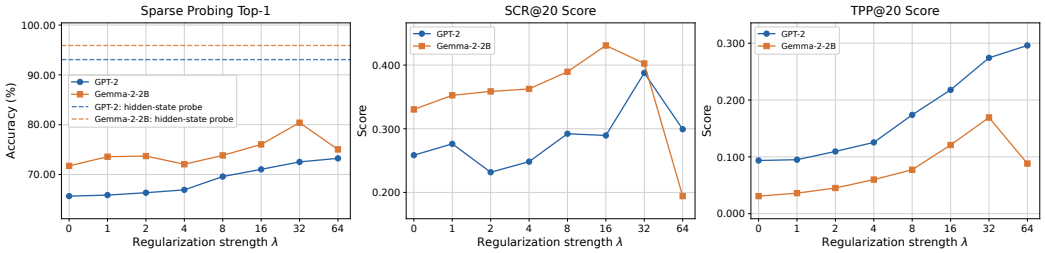

Figure 3: **SAEBench evaluation.** Sparse probing Top-1 accuracy, SCR@20, and TPP@20 across regularization strengths for TopK SAEs with $K = 64$ and $F = 16{,}384$ features, trained on GPT-2 (Radford et al., 2019) layer 7 and Gemma-2-2B (Gemma Team, 2024) layer 12 activations. Dashed lines indicate hidden-state probe accuracy, and $\lambda = 0$ denotes the unregularized baseline.

where S is the number of runs, $e_i^{(a)}$ is a unit-normalized decoder direction, and $\theta = 0.7$. Thus, a feature reappears when another run contains a sufficiently similar decoder direction. We summarize dictionary-level reproducibility by

$$\bar{p}^{(a)} = \frac{1}{F} \sum_{i=1}^{F} \hat{p}_i^{(a)},$$

where F is the dictionary size.

The left panel of Figure 4 shows a non-monotonic dependence on regularization strength. For both GPT-2 (Radford et al., 2019) layers, moderate kNN regularization increases mean reappearance probability relative to the baseline, with a larger improvement at layer 7, while reducing explained variance. Gemma-2-2B (Gemma Team, 2024) layer 7 likewise exhibits an intermediate maximum among the displayed regularized settings. Across all three configurations, increasing regularization beyond this regime reduces both reproducibility and reconstruction quality. The beneficial strength therefore depends on the representation being analyzed.

The right panel of Figure 4 reveals that the improvement is concentrated at the extremes of the reappearance distribution, rather than being merely an increase in its mean. For GPT-2 (Radford et al., 2019) layer 7, kNN regularization at $\lambda = 8$ nearly halves the number of features in the lowest-reappearance bin and increases the number in the highest-reappearance bin, while the intermediate bins remain comparatively unchanged. At the distribution level, this is a redistribution of mass from the least reproducible to the most reproducible feature population. The regularized dictionary therefore contains fewer strongly seed-dependent directions and a larger core of consistently recovered features.

A plausible mechanism is reduced ambiguity in the encoder's local feature selection. Nearby hidden states can activate different, nearly collinear decoder atoms, producing similar reconstructions but sharply different sparse codes. Reconstruction loss alone weakly penalizes such switches. The kNN objective additionally matches local code distances to input distances up to a shared scale, discouraging disproportionate changes while preserving nonzero differences. This constrains the encoder's finite differences, including pairs with different TopK supports.

5.4 Adversarial Robustness

We continue our evaluation by examining whether kNN SAE representations remain stable under deliberately chosen input perturbations. Following (Li et al., 2026), we use untargeted population attacks that seek to change the active feature set without prescribing a target representation.

We evaluate TopK SAEs on GPT-2 (Radford et al., 2019) layer 7/10 and Gemma-2-2B (Gemma Team, 2024) layer 12/18, varying batch kNN regularization at fixed sparsity. The attack either replaces one token in the prompt while leaving the others unchanged, or adds a single token at the end of the original prompt. We tailor the attack to each SAE separately, allowing the same number of attempts to find an edit that disrupts its active feature set as much as possible.

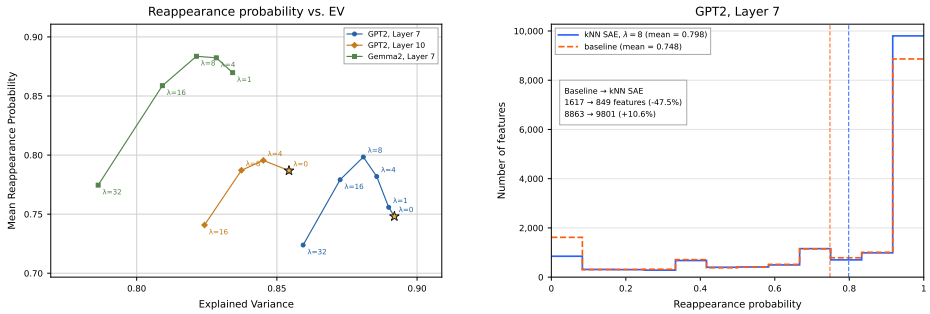

Figure 4: **Feature reproducibility across training seeds.** Left: the trade-off between feature reproducibility and reconstruction quality across regularization strengths. Right: regularization shifts more features toward consistent recovery across seeds, illustrated for GPT-2 (Radford et al., 2019) layer 7 with $\lambda = 8$. Vertical lines indicate mean reappearance probabilities.

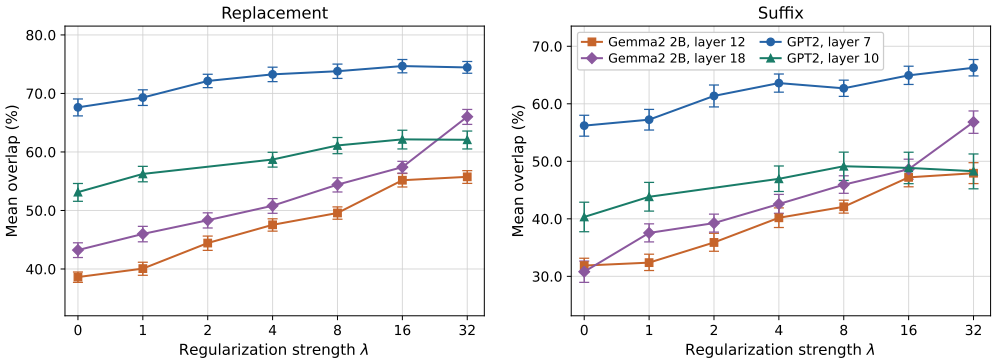

Figure 5: **SAE robustness under untargeted attacks**. Left: mean retained feature overlap under token replacement attacks across regularization strengths λ. Right: the same metric under single-token suffix attacks. Results are shown for Gemma-2-2B at layers 12 and 18 ($F = 32{,}768$) and GPT-2 at layers 7 and 10 ($F = 16{,}384$), with $K = 64$ throughout. Error bars indicate pointwise 95% confidence intervals from bootstrap resampling clustered by source text. Higher overlap indicates greater robustness.

For an original prompt x and its adversarially modified version x^{adv}, we extract the last-token hidden state at the evaluated layer and encode it with the SAE. Let $S_K(x)$ and $S_K(x^{\text{adv}})$ denote the resulting sets of K active features. Our goal is to reduce the sensitivity of SAE representations to small input edits, so that changing a single prompt token does not cause a disproportionate change in the active feature set. We measure the fraction of original features retained after attack:

$$O_K(x, x^{\text{adv}}) = \frac{|S_K(x) \cap S_K(x^{\text{adv}})|}{K}.$$

Higher overlap indicates greater retention. Error bars show pointwise 95% confidence intervals from cluster bootstrapping over original texts.

As shown in Figure 5, both models retain more of their original active features with kNN regularization than without it. Since sparsity is fixed, higher overlap means that the attack replaces fewer features in the sparse description. This benefit appears under both perturbation types and generally increases with regularization strength, although the relationship is not strictly monotonic. Suffix attacks leave less overlap in the displayed comparisons, showing that an appended token can substantially alter the representation even while preserving the original text.

The regularizer constrains distances between codes, whereas this experiment tests whether feature identities survive adversarial edits. Improved retention therefore connects geometric regularization to more stable feature selection, including against attacks optimized specifically for each SAE.

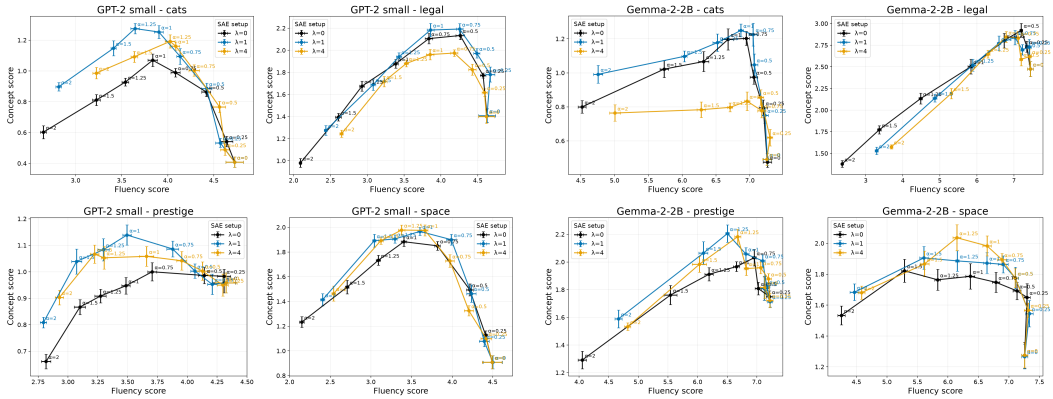

Figure 6: **Concept steering versus generation quality.** Steering results for GPT-2 small at layer 7 (left) and Gemma-2-2B at layer 18 (right). Black, blue, and orange denote $\lambda = 0$, $\lambda = 1$, and $\lambda = 4$, respectively. Points are labeled by the steering coefficient α; movement upward indicates stronger expression of the target concept, while movement to the left indicates loss of fluency. Coordinates are means over three independently selected features, 12 prompts, and 10 stochastic sample replicates. Error bars show SEM over replicate-level aggregates. The four concepts are selected qualitative case studies rather than an unbiased sample from a predefined concept population.

5.5 Causal steering with SAE features

We next evaluate whether the features learned by kNN SAE exert a more controlled causal influence on model generations. We study four concepts—*cats*, *legal*, *prestige*, and *space*—using GPT-2 small and Gemma-2-2B. For each model, we compare an unregularized TopK SAE ($\lambda = 0$) with kNN SAEs trained using $\lambda \in \{1, 4\}$. For every SAE and concept, we independently select the three features that activate most selectively on concept-relevant tokens in a one-million-token FineWeb activation bank. The experiment therefore compares the steering behavior supported by each learned dictionary under a common feature-discovery procedure; it does not assume that feature identities are matched across SAEs.

For a selected feature f, we intervene along its unit-normalized decoder direction $\mathbf{d}_f$. We calibrate the intervention using the largest activation of that feature in the discovery bank,

$$\mathbf{h} \leftarrow \mathbf{h} + \alpha m_f \mathbf{d}_f, \qquad m_f = \max_x z_f(\mathbf{h}_x), \qquad \|\mathbf{d}_f\|_2 = 1.$$

The intervention is applied at the target residual-stream layer at every autoregressive generation step. Thus, α specifies the intervention strength relative to the feature's empirical activation scale, rather than as an uncalibrated displacement in hidden-state space.

We sweep $\alpha \in \{0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2\}$ and generate continuations from 12 concept-neutral prompts. An LLM judge assigns each continuation two scores from 0 to 10: a *concept score*, measuring how strongly the target concept appears, and a *fluency score*, measuring linguistic coherence and continuity. Full feature-selection, generation, judging, and aggregation details are provided in Appendix C.

Figure 6 shows a consistent qualitative pattern. Starting from the unsteered model, increasing α initially strengthens the target concept while gradually reducing fluency. At sufficiently large strengths, however, the intervention damages generation quality and often ceases to improve—or even reduces—the concept score. Steering quality must therefore be evaluated as a two-dimensional trade-off: the desired intervention produces stronger concept expression without unnecessarily disrupting the rest of the generation.

Across the four GPT-2 case studies, the kNN SAE trajectories generally define a more favorable trade-off than the unregularized TopK SAE. At comparable concept scores, the $\lambda \in \{1, 4\}$ interventions tend to retain higher fluency; equivalently, at comparable fluency, they often induce the target concept more strongly. The same pattern appears for Gemma-2-2B in the *legal*, *prestige*, and especially *space* cases, most clearly once the intervention becomes strong. The result is not universal: for

Gemma's *cats* concept, $\lambda = 1$ is competitive with the baseline, whereas $\lambda = 4$ yields substantially weaker concept steering. Thus, stronger neighborhood regularization is not monotonically better for every learned feature.

In these case studies, kNN SAE steering often retains higher fluency at comparable concept scores. We hypothesize that this advantage arises from better preservation of activation geometry. Additive steering can move hidden states outside the distribution encountered during ordinary inference, disrupting downstream computation and degrading generation quality (Rodríguez et al., 2025); interventions that follow the activation manifold have correspondingly been shown to produce more natural behavior than linear paths through off-manifold regions (Wurgaft et al., 2026). Our regularizer does not constrain steering trajectories directly. It preserves local distances between hidden states in $\mathcal{H}$ and sparse codes in $\mathcal{Z}$, while the jointly trained decoder maps those codes back to $\hat{\mathcal{H}}$. This may produce decoder directions that better align with locally supported variation in hidden-state space. The trade-offs in Figure 6 are consistent with this explanation, but we do not measure the distributional shift of the steered activations directly, so the proposed mechanism remains a hypothesis.

6 Conclusion

We introduced kNN SAE, which augments sparse autoencoder training with a CkNN-based objective that matches local distances between model activations and their sparse codes up to a shared scale. At matched sparsity, this constraint improves neighborhood preservation across three language models and yields gains in concept detection, targeted ablation, and resistance to adversarial feature disruption on GPT-2 (Radford et al., 2019) and Gemma-2-2B (Gemma Team, 2024). Moderate regularization also improves feature reproducibility across independent GPT-2 (Radford et al., 2019) training runs, while selected steering case studies suggest more favorable concept–fluency trade-offs. These benefits depend on regularization strength and come at a cost in reconstruction fidelity. Together, our results support local geometry preservation as a useful complement to reconstruction and sparsity for learning more reproducible and practically useful SAE features.

Limitations. Our evaluation covers three language models and selected layers, with downstream, robustness, and steering experiments restricted to GPT-2 (Radford et al., 2019) and Gemma-2-2B (Gemma Team, 2024). Generalization to larger models, other SAE architectures, and broader data distributions remains to be established. Regularization introduces training overhead, and excessive strength can reduce both reconstruction quality and feature reproducibility. The steering results concern four selected concepts and do not establish a general advantage or verify the proposed geometric mechanism. Finally, improved reproducibility does not establish identifiability of individual features. Future work should investigate how to select regularization strength for a target application and when preserved local geometry translates into reliable feature-level interpretations.

AI Use Statement

We used an AI coding assistant to edit existing code and write new code for our experiments. We also used it to edit and polish the manuscript at the paragraph level. We did not use AI to generate ideas or conduct literature searches. We take full responsibility for the content of the paper.

References

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. URL `https://transformer-circuits.pub/2023/monosemantic-features/index.html`.

Nutan Chen, Patrick van der Smagt, and Botond Cseke. Local distance preserving auto-encoders using continuous kNN graphs. In Alexander Cloninger, Timothy Doster, Tegan Emerson, Manohar

Kaul, Ira Ktena, Henry Kvinge, Nina Miolane, Bastian Rieck, Sarah Tymochko, and Guy Wolf (eds.), *Proceedings of Topological, Algebraic, and Geometric Learning Workshops 2022*, volume 196 of *Proceedings of Machine Learning Research*, pp. 55–66. PMLR, 25 Feb–22 Jul 2022. URL https://proceedings.mlr.press/v196/chen22b.html.

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

Andres F. Duque, Sacha Morin, Guy Wolf, and Kevin R. Moon. Geometry regularized autoencoders. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7381–7394, 2023. doi: 10.1109/TPAMI.2022.3222104. URL https://doi.org/10.1109/TPAMI.2022.3222104.

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

Gemma Team. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. doi: 10.48550/arXiv.2408.00118. URL https://arxiv.org/abs/2408.00118.

Gleb Gerasimov, Timofei Rusalev, Nikita Balagansky, Daniil Laptev, Vadim Kurochkin, and Daniil Gavrilov. Unstable features, reproducible subspaces: Understanding seed dependence in sparse autoencoders. *arXiv preprint arXiv:2606.12138*, 2026. URL https://arxiv.org/abs/2606.12138.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Isaac Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum Stuart 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. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29223–29264. PMLR, 2025. URL https://proceedings.mlr.press/v267/karvonen25a.html.

Yonghyeon Lee, Hyeokjun Kwon, and Frank C. Park. Neighborhood reconstructing autoencoders. In *Advances in Neural Information Processing Systems*, volume 34, pp. 536–546, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/05311655a15b75fab86956663e1819cd-Abstract.html.

Aaron J. Li, Suraj Srinivas, Usha Bhalla, and Himabindu Lakkaraju. Evaluating adversarial robustness of concept representations in sparse autoencoders. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5940–5957, Rabat, Morocco, March 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.eacl-long.279. URL https://aclanthology.org/2026.eacl-long.279/.

Jungbin Lim, Jihwan Kim, Yonghyeon Lee, Cheongjae Jang, and Frank C. Park. Graph geometry-preserving autoencoders. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 29795–29815. PMLR, 2024. URL https://proceedings.mlr.press/v235/lim24a.html.

Meta. Llama 3.2 model card, 2024. URL https://github.com/meta-llama/llama-models/blob/main/models/llama3_2/MODEL_CARD.md. Accessed September 26, 2026.

Michael Moor, Max Horn, Bastian Rieck, and Karsten Borgwardt. Topological autoencoders. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7045–7054. PMLR, 2020. URL https://proceedings.mlr.press/v119/moor20a.html.

Favour Nerrise, Lucy Yin, Mohammad H. Abbasi, Kilian M. Pohl, and Ehsan Adeli. GeoSAE: Geometric prior-guided layer-wise sparse autoencoder annotation of brain MRI foundation models. In *Proceedings of the CVPR Workshop on Computer Vision for Real-world Clinical Translation (CV4Clinic)*, 2026. URL https://arxiv.org/abs/2605.01829.

Gonçalo Paulo and Nora Belrose. Sparse autoencoders trained on the same data learn different features. *arXiv preprint arXiv:2501.16615*, 2025. URL https://arxiv.org/abs/2501.16615.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Pau Rodríguez, Arno Blaas, Michal Klein, Luca Zappella, Nicholas Apostoloff, Marco Cuturi, and Xavier Suau. Controlling language and diffusion models by transporting activations. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=l2zFn6TIQi.

Daniel Wurgaft, Can Rager, Matthew Kowal, Vasudev Shyam, Sheridan Feucht, Usha Bhalla, Tal Haklay, Eric Bigelow, Raphael Sarfati, Thomas McGrath, Owen Lewis, Jack Merullo, Noah Goodman, Thomas Fel, Atticus Geiger, and Ekdeep Singh Lubana. Manifold steering reveals the shared geometry of neural network representation and behavior. *arXiv preprint arXiv:2605.05115*, 2026. URL https://arxiv.org/abs/2605.05115.

Jehyeok Yeon, Federico Cinus, Yifan Wu, and Luca Luceri. Graph-regularized sparse autoencoders for LLM safety steering. *arXiv preprint arXiv:2512.06655*, 2025. URL https://arxiv.org/abs/2512.06655v3.

A Training Setup

Training data and activations. Unless otherwise specified, we train on `resid_post` activations from GPT-2 small layer 7, Gemma-2-2B layer 12, and Llama-3.2-1B layer 10. The language models remain frozen. Activations are extracted online from the `sample-10BT` subset of FineWeb using sequences of 1,024 tokens and a model batch size of 16. We assign documents deterministically by content hash to training, validation, and calibration partitions in a 98:1:1 ratio. The activation buffer stores 32 SAE batches and is therefore refilled approximately once every 32 optimization steps.

Before encoding, each hidden state is centered and scaled by its standard deviation across the residual dimension. We use TopK SAEs with $F = 16{,}384$ features and target sparsity $K = 64$. During the first 10% of training, K decreases linearly from 128 to 64. Encoder weights use Kaiming-uniform initialization; decoder directions are initialized from the transposed encoder weights and normalized to unit Euclidean norm. We project out the decoder-gradient component parallel to each decoder direction and renormalize the directions after every optimizer update.

The base TopK objective combines mean-squared reconstruction error with the auxiliary loss for inactive features from Gao et al. (2024). A feature is treated as inactive after ten consecutive SAE batches; the auxiliary reconstruction uses up to 512 inactive features with coefficient $1/32$. We train with an SAE batch size of 8,192 for one billion tokens, corresponding to 122,070 optimization steps. We use AdamW with learning rate 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.99$, weight decay 0.05, and gradient-norm clipping at 1. The learning rate warms up for the first 10% of training and then follows a cosine schedule down to 10^{-6}. Activation extraction and SAE optimization use float32. We use seed 42 except in experiments that explicitly vary the initialization seed.

kNN regularization. For the batch-local kNN objective, we construct a CkNN graph from the 8,192 normalized activations in the current SAE batch, using $k = 16$ and $\delta = 1$. We sample 256 anchors uniformly from the non-isolated graph vertices and four graph neighbors per anchor to estimate the geometry loss. Distances in $\mathcal{Z}$ are computed directly from the sparse TopK codes. We estimate the closed-form distance scale from the sampled edges using exponential moving averages of its sufficient statistics with decay 0.99.

Full-run amortized cost

Model	Activation generation (ms/step)	Neighborhood (ms/step)	Total overhead (ms/step)	Slowdown	Additional training time
GPT-2 small, layer 7	34.137	7.224	7.043	14.81%	14.33 min
Gemma-2-2B, layer 12	142.680	10.153	10.864	6.58%	22.10 min

Cost during the geometry-active phase

Model	Neighborhood (ms/active step)	Total overhead (ms/active step)
GPT-2 small, layer 7	10.320	10.062
Gemma-2-2B, layer 12	14.505	15.520

Table 1: **Training overhead of kNN SAE.** Matched TopK and kNN SAE timing measurements on one NVIDIA H100 80GB with $F = 16{,}384$, $K = 64$, and an SAE batch size of 8,192. The upper table reports costs amortized over the complete training schedule: because geometry is active only during the final 70% of training, these values are 0.7 times the corresponding active-phase measurements. Slowdown is measured relative to the matched TopK core training time, including amortized online activation extraction. Additional training time is extrapolated over the 85,449 geometry-active steps of a 122,070-step run.

The geometry loss is disabled during the first 30% of training. It is evaluated at every optimization step thereafter, giving 85,449 geometry-active steps, and its coefficient is linearly increased from zero to the target λ over the first 5,000 active steps. The unregularized $\lambda = 0$ runs skip graph construction and geometry-loss computation entirely. All other architecture, data, optimization, and activation-buffer settings are matched between the TopK and kNN SAE runs.

Training efficiency. We benchmark the $\lambda = 4$ configuration against the matched TopK baseline on one NVIDIA H100 80GB. Activation generation is identical between the two methods and is excluded from the kNN-overhead numerator. It remains part of the baseline time used to calculate slowdown, since online activation extraction contributes to total training time. The long-run amortized activation-generation costs are 34.137 ms per step for GPT-2 and 142.680 ms per step for Gemma-2-2B. We exclude one-time CUDA compilation outliers and subtract the approximately 0.003 ms of empty CUDA-event bookkeeping recorded as neighborhood time for the TopK baseline.

For GPT-2, the measured total overhead is slightly smaller than the time attributed to neighborhood construction alone. This small negative difference is caused by measurement noise and execution-order effects between otherwise matched runs. For Gemma-2-2B, the approximately one millisecond per active step beyond neighborhood construction is primarily due to backpropagation through the geometry loss. The absolute additional time is larger for Gemma because its active-step overhead is larger, while its percentage slowdown is smaller because generating Gemma activations is substantially more expensive.

A.1 Optimal Gamma Derivation

For a fixed batch graph G_b, define

$$d_{ij}^h = \|\mathbf{h}_i - \mathbf{h}_j\|_2, \qquad d_{ij}^z = \|\mathbf{z}_i - \mathbf{z}_j\|_2.$$

The CkNN objective as a function of γ is

$$\mathcal{L}_{\mathrm{CkNN}}(\gamma) = \frac{1}{|E_b|} \sum_{(i,j)\in E_b} \left(s_b^{-1} d_{ij}^h - \gamma d_{ij}^z\right)^2 .$$

Since this is a one-dimensional least-squares problem, its optimum is obtained by differentiating with respect to γ:

$$\frac{\partial \mathcal{L}_{\mathrm{CkNN}}}{\partial \gamma} = -\frac{2}{|E_b|} \sum_{(i,j)\in E_b} d_{ij}^z \left(s_b^{-1} d_{ij}^h - \gamma d_{ij}^z\right).$$

Setting the derivative to zero gives

$$\sum_{(i,j)\in E_b} s_b^{-1} d_{ij}^h d_{ij}^z = \gamma \sum_{(i,j)\in E_b} \left(d_{ij}^z\right)^2,$$

and therefore

$$\gamma^* = \frac{\sum_{(i,j)\in E_b} d_{ij}^h d_{ij}^z}{s_b \sum_{(i,j)\in E_b} \left(d_{ij}^z\right)^2}.$$

Substituting the definitions of the pairwise distances yields

$$\boxed{\gamma^* = \frac{\sum_{(i,j)\in E_b} \|\mathbf{h}_i - \mathbf{h}_j\|_2 \cdot \|\mathbf{z}_i - \mathbf{z}_j\|_2}{s_b \sum_{(i,j)\in E_b} \|\mathbf{z}_i - \mathbf{z}_j\|_2^2}}.$$

Thus, γ^* is simply the least-squares scaling coefficient aligning latent-space distances with the normalized distances in the original activation space.

B STRUCTURED NUMERICAL-ROLE BENCHMARK

Data construction. We collect examples from the `sample-10BT` configuration of FineWeb using high-precision patterns for six numerical roles: currency amount, clock time, calendar date, sports score, citation field, and product specification. The target is respectively the number following a currency symbol, the minute field of a time, the day field of a date, the second value in a sports score, a page or volume number, or the value following a product-attribute marker. Sports and citation candidates are additionally required to contain role-specific contextual terms. The extraction pass scans 233,147 documents.

To remove numerical identity as a shortcut, each extracted value is replaced with an integer between 10 and 28. These values are approximately uniform within each role, with counts differing by at most one. The six roles are exactly balanced. Near-duplicate contexts are removed using a normalized text signature in which all numbers are replaced by a placeholder.

Documents are assigned to discovery and holdout splits by hashing the source document before extracting examples. Fragments from the same document therefore cannot occur in both splits. The discovery set contains 400 examples per role (2,400 in total), and the holdout set contains 250 examples per role (1,500 in total).

Representations. For each example, we extract the `resid_post` activation from layer 18 of Gemma-2-2B (Gemma Team, 2024) at the first token of the target number. Since Gemma is causal, this activation depends only on the beginning of the target number and its preceding context. We compare the original activation $\mathbf{h}_i \in \mathcal{H}$ with codes $\mathbf{z}_i \in \mathcal{Z}$ from a vanilla TopK SAE and a kNN-regularized TopK SAE with $\lambda = 4$. Both SAEs have 16,384 latent features and exactly 64 active features per input. The kNN SAE uses the batch-local CkNN objective with $k = 16$ and $\delta = 1$.

Neighborhood classification. We evaluate whether each representation retains the contextual role of the target number using a 5-nearest-neighbor classifier. For a holdout example i and representation space $\mathcal{X} \in \{\mathcal{H}, \mathcal{Z}\}$, let $\mathcal{N}_5^{\mathcal{X}}(i; \mathcal{D}_{\text{disc}})$ be its five nearest discovery examples under Euclidean distance. Its predicted role is

$$\hat{y}_i^{\mathcal{X}} = \text{mode}\left\{y_j : j \in \mathcal{N}_5^{\mathcal{X}}(i; \mathcal{D}_{\text{disc}})\right\}, \tag{1}$$

and accuracy is computed over all $N = 1{,}500$ holdout examples:

$$\text{Acc}_5(\mathcal{X}) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\hat{y}_i^{\mathcal{X}} = y_i\right]. \tag{2}$$

Model	Intervention site	Residual width	Dictionary size
GPT-2 small	layer 7 `resid_post`	768	16,384
Gemma-2-2B	layer 18 `resid_post`	2,304	16,384

Table 2: Models and residual-stream intervention sites used in the steering evaluation.

Neighborhood preservation. For each holdout example, we independently compute its 16 nearest neighbors among the other holdout examples in $\mathcal{H}$ and in the SAE latent space $\mathcal{Z}$. Recall@16 is the mean fraction of hidden-state neighbors retained after SAE encoding:

$$\mathrm{Recall@16}(\mathcal{Z};\mathcal{H}) = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\mathcal{N}_{16}^{\mathcal{H}}(i)\cap\mathcal{N}_{16}^{\mathcal{Z}}(i)\right|}{16}. \tag{3}$$

All distances used for 5-NN accuracy and Recall@16 are computed in the original full-dimensional representation spaces.

PCA visualization. PCA is fitted separately to all $1{,}500$ holdout representations in each space. The first two components are shown for the same stratified subset of 100 examples per role, giving 600 displayed points. Each SAE projection is centered, normalized to a common global scale, and aligned to the projection of $\mathcal{H}$ using an orthogonal Procrustes transformation. This alignment permits only rotation or reflection and does not alter distances within a projection.

As a quantitative summary of the two-dimensional geometry, we compute the Spearman correlation between all pairwise distances in the PCA projection of $\mathcal{H}$ and the corresponding distances in each SAE projection:

$$\rho_{\mathrm{PCA}}(\mathcal{Z},\mathcal{H}) = \rho_{\mathrm{S}}\left(\left\{\left\|\mathbf{p}_i^{\mathcal{H}}-\mathbf{p}_j^{\mathcal{H}}\right\|_2\right\}_{i<j},\left\{\left\|\mathbf{p}_i^{\mathcal{Z}}-\mathbf{p}_j^{\mathcal{Z}}\right\|_2\right\}_{i<j}\right). \tag{4}$$

Results. The hidden-state representation reaches a 5-NN accuracy of 0.961, confirming that the task is strongly represented in $\mathcal{H}$. The vanilla TopK SAE obtains an accuracy of 0.947, Recall@16 of 0.743, and PCA distance correlation of 0.720. The kNN SAE with $\lambda = 4$ improves these values to 0.958, 0.826, and 0.773, respectively. Thus, kNN regularization recovers most of the classification performance lost during SAE encoding while preserving substantially more of the original local neighborhoods.

C Steering evaluation details

Models and SAEs. We evaluate GPT-2 small at layer 7 `resid_post` and Gemma-2-2B at layer 18 `resid_post`. For each model, we compare independently trained TopK SAEs with neighborhood-regularization strengths $\lambda \in \{0, 1, 4\}$. All SAEs have 16,384 features, TopK sparsity $K = 64$, and training seed 42. We use folded checkpoints in which every decoder direction has unit Euclidean norm.

Activation bank and concept labels. We use one million token-level residual-stream activations from the `sample-10BT` split of FineWeb for both feature discovery and intervention-scale calibration. For an activation $\mathbf{h}_x$, the corresponding SAE code is

$$\mathbf{z}_x = \mathrm{SAE.encode}(\mathbf{h}_x).$$

We label a token position as relevant to concept c when its decoded, lower-cased, and boundary-normalized token exactly matches an entry in the concept vocabulary. Only the concept-token position is labeled; no surrounding context window is included. We use the following vocabularies:

cats: `cat, cats, kitten, kittens, kitty, kitties, feline, felines, tomcat, tabby, purr, purring, meow, meowing, whisker, whiskers.`

legal: `law, laws, legal, legally, court, courts, judge, judges, trial, trials, lawyer, lawyers, attorney, attorneys,`

Model	Concept	$\lambda = 0$	$\lambda = 1$	$\lambda = 4$
GPT-2 small	cats	1191, 8493, 12222	8493, 12222, 12226	6454, 8493, 12226
GPT-2 small	legal	7828, 9959, 10474	1344, 7828, 10474	1344, 7828, 10474
GPT-2 small	prestige	5029, 9250, 15118	5029, 8859, 9250	5029, 8859, 9250
GPT-2 small	space	2017, 8337, 13000	2017, 8337, 13000	2017, 8337, 13000
Gemma-2-2B	cats	5684, 8353, 10582	5684, 8353, 10582	8353, 10582, 12235
Gemma-2-2B	legal	207, 4988, 6339	207, 4988, 6339	207, 4988, 6339
Gemma-2-2B	prestige	10388, 10461, 14617	10388, 10461, 14617	10388, 10461, 14617
Gemma-2-2B	space	2587, 7112, 10090	2587, 7112, 10090	2587, 7112, 10090

Table 3: Features selected independently for each model, concept, and regularization strength.

`lawsuit, lawsuits, verdict, verdicts, defendant, defendants, plaintiff, plaintiffs, evidence, testimony, witness, witnesses, criminal, crime, crimes, offense, offences, sentence, sentenced, appeal, appeals, constitution, contract, contracts, justice.`

prestige: `prestige, prestigious, luxury, luxurious, elite, exclusive, premium, designer, couture, opulent, opulence, glamour, glamorous, wealth, wealthy, affluent, expensive, lavish, jewelry, jewellery, diamond, diamonds, rolex, gucci, chanel, prada, ferrari, yacht, yachts, penthouse.`

space: `planet, planets, star, stars, galaxy, galaxies, moon, moons, mars, jupiter, saturn, rocket, rockets, orbit, orbital, telescope, telescopes, nebula, nebulae, astronaut, astronauts, cosmos, universe, solar, lunar, spaceship, spaceships, spacecraft, asteroid, asteroids, comet, comets.`

Feature discovery. Let R_c denote the token positions relevant to concept c, and let B_c contain all remaining positions. For every feature f, we compute its mean activation and activation frequency on the relevant and background sets:

$$\mu_{\mathrm{rel}}(f,c) = \frac{1}{|R_c|}\sum_{x\in R_c} z_{x,f}, \qquad \mu_{\mathrm{bg}}(f,c) = \frac{1}{|B_c|}\sum_{x\in B_c} z_{x,f},$$

$$p_{\mathrm{rel}}(f,c) = \frac{1}{|R_c|}\sum_{x\in R_c} \mathbf{1}[z_{x,f}>0], \quad p_{\mathrm{bg}}(f,c) = \frac{1}{|B_c|}\sum_{x\in B_c} \mathbf{1}[z_{x,f}>0].$$

We rank features using the selectivity gap

$$g(f,c) = \mu_{\mathrm{rel}}(f,c) - \mu_{\mathrm{bg}}(f,c).$$

To exclude broad features that activate frequently in unrelated contexts, a feature is eligible only if

$$p_{\mathrm{bg}}(f,c) \le 0.01, \qquad p_{\mathrm{rel}}(f,c) \ge 3p_{\mathrm{bg}}(f,c), \qquad g(f,c) > 0.$$

For every model, regularization strength, and concept, we retain the three eligible features with the largest selectivity gaps. Selection is performed independently for each SAE. Consequently, the evaluation compares the features supported by each dictionary under the same discovery rule rather than forcing a correspondence between latent indices. For qualitative auditing, we also retain the 30 highest-activation FineWeb contexts for each selected feature, including 20 tokens on either side of the activated position.

Intervention construction. Let $\mathbf{d}_f$ denote the folded decoder direction of feature f, normalized such that $\|\mathbf{d}_f\|_2 = 1$. We calibrate the intervention using

$$m_f = \max_x z_{x,f},$$

where the maximum is taken over the same one-million-token activation bank used for feature discovery. For steering coefficient α, the intervention is

$$\mathbf{v}(f,\alpha) = \alpha m_f \mathbf{d}_f, \qquad \mathbf{h} \leftarrow \mathbf{h} + \mathbf{v}(f,\alpha).$$

The vector is added at the selected residual-stream hook at every generation step, but not while processing the original prompt. Since the decoder direction has unit norm,

$$\|\mathbf{v}(f, \alpha)\|_2 = \alpha m_f.$$

Thus, α measures intervention strength in multiples of the feature's largest observed activation; it is not a common absolute displacement across features. We evaluate

$$\alpha \in \{0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2\}.$$

The $\alpha = 0$ condition applies no intervention and is shared across the three features within a run.

Generation protocol. All conditions use the same 12 concept-neutral continuation prompts:

1. The old machine began to make a strange sound, and
2. At the edge of the city, a group of friends decided
3. The researcher opened the sealed box and
4. The meeting ended quietly, but the report said
5. After the rain stopped, the driver looked down the empty road and
6. The small village woke before sunrise, while the shopkeeper
7. During the afternoon session, the committee examined the proposal and
8. The child picked up the letter from the table and
9. Near the railway station, an unfamiliar man asked the guard
10. The architect spread the old plans across the desk and
11. When the lights went out, everyone in the room
12. The historian compared the two accounts and concluded that

The prompts contain no concept-specific instructions. Each continuation contains at most 96 newly generated tokens and terminates earlier at EOS. We sample from the full softmax distribution with temperature 1, without top-k or nucleus truncation. GPT-2 inputs do not include a BOS token, whereas Gemma inputs use the BOS token prescribed by its tokenizer.

For every positive $(\text{model}, \lambda, \text{concept}, f, \alpha, \text{prompt})$ condition, we collect 10 stochastic samples. These samples are produced by two five-sample runs initialized with generation seeds 1729 and 918273. The prompt index is incorporated into the random stream, and the second run's sample indices are shifted from $0, \ldots, 4$ to $5, \ldots, 9$ when the results are merged. The 10 samples are therefore stochastic generation replicates, not independently configured experiment-level seeds.

LLM-as-a-judge evaluation. We judge generations using Qwen3-32B served locally through vLLM in bfloat16, with judge temperature zero. The evaluator receives the target concept, the original prompt, and the complete generated continuation:

```
Evaluate this generation.  Target concept:  {concept}
Original prompt:  {prompt}
Generation:  {generation}
Return only JSON with integer fields coherence and concept,
each 0--10.
Coherence measures fluency and logical continuity.  Concept
measures how strongly the target concept is present,
regardless of whether it is named directly.
```

The returned `coherence` value is reported as the *fluency score*. The *concept score* captures both explicit and implicit expression of the target concept. We retain the raw judge responses for auditing and use the same judge, evaluator prompt, and decoding configuration for every condition.

Aggregation and uncertainty. Prompts and selected features are averaged within each stochastic replicate; they are not treated as independent experimental repetitions. For a positive steering coefficient, let $y_{f,p,s}$ denote either score for feature f, prompt p, and stochastic replicate s. We first compute

$$r_s = \frac{1}{3 \cdot 12} \sum_{f=1}^{3} \sum_{p=1}^{12} y_{f,p,s},$$

and report

$$\bar{r} = \frac{1}{10} \sum_{s=1}^{10} r_s.$$

For $\alpha = 0$, there is no feature dimension, so r_s is the mean over the 12 prompts. Baseline points may be displayed separately for the three λ runs even though the underlying unsteered language model is unchanged.

Horizontal and vertical error bars are computed separately from the replicate-level fluency and concept aggregates. Both show the standard error of the mean:

$$\text{SEM} = \frac{\sqrt{\frac{1}{9} \sum_{s=1}^{10} (r_s - \bar{r})^2}}{\sqrt{10}}.$$

The error bars therefore use 10 independent stochastic replicates rather than treating the $3 \times 12 \times 10 = 360$ feature–prompt–sample observations at each positive α as independent.

For each concept and λ, the unsteered condition contains $12 \times 10 = 120$ judged generations. Every positive value of α contains $3 \times 12 \times 10 = 360$ generations. The seven positive coefficients therefore contribute 2,520 generations, giving 2,640 judged generations per concept and λ, 10,560 per model and λ, and 63,360 judged generations across both models and all three regularization strengths.

Scope and limitations. The plotted curves connect conditions in increasing order of α; they are empirical fluency–concept trade-offs, not temporal trajectories. A favorable condition must either increase concept expression at comparable fluency or preserve fluency at comparable concept strength. Concept score alone is not sufficient.

The four concepts were selected from a larger exploratory set and should be interpreted as qualitative case studies. Because feature discovery is performed independently for each SAE, the experiment evaluates the steering behavior supported by each learned dictionary rather than providing a matched-direction causal comparison. Calibration by a maximum activation may also be sensitive to rare, high-activation examples. Finally, the shared judge supports controlled relative comparisons but does not eliminate judge-specific biases. In particular, this experiment does not directly estimate the density or out-of-distribution distance of the intervened hidden states; the geometric explanation discussed in Section 5.5 should therefore be treated as a hypothesis motivated by the observed fluency–concept trade-off.