DOES HIGHER INTERPRETABILITY IMPLY BETTER UTILITY? A PAIRWISE ANALYSIS ON SPARSE AUTOENCODERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse Autoencoders (SAEs) are widely used to steer large language models (LLMs), based on the assumption that their interpretable features naturally enable effective model behavior steering. Yet, a fundamental question remains unanswered: *does higher interpretability indeed imply better steering utility?* To answer this question, we train 90 SAEs across three LLMs (Gemma-2-2B, Qwen-2.5-3B, Gemma-2-9B), spanning five architectures and six sparsity levels, and evaluate their interpretability and steering utility based on SAEBENCH (Karvonen et al., 2025) and AXBENCH (Wu et al., 2025) respectively, and perform a rank-agreement analysis via Kendall's rank coefficients τ_b. Based on the framework, Our analysis reveals only a relatively weak positive association ($\tau_b \approx 0.298$), indicating that interpretability is an insufficient proxy for steering performance. We conjecture the interpretability-utility gap may stem from the selection of SAE features as not all of them are equally effective for steering. To further find features that truly steer the behavior of LLMs, we propose a novel selection criterion: Δ *Token Confidence*, which measures how much amplifying a feature changes the next token distribution. We show that our method improves the steering performance of three LLMs by **52.52%** compared to the current best output score-based criterion (Arad et al., 2025). Strikingly, after selecting features with high Δ *Token Confidence*, the correlation between interpretability and utility vanishes ($\tau_b \approx 0$), and can even become negative. This further highlights the divergence between interpretability and utility for the most effective steering features.

1 INTRODUCTION

As Large Language Models (LLMs) become more widely used in real-world applications, ensuring the safety of their outputs is increasingly important (Kumar et al., 2024; Ji et al., 2023; Inan et al., 2023). Reliable and controllable behavior is essential for deploying these LLMs in more situations (Chen et al., 2024). Fine-tuning is the standard way to improve controllability, but it requires labeled data, significant training time, and compute resources (Hu et al., 2022; Wang et al., 2025c). This has spawned a series of representation-based interventions, i.e., steering, that guide LLM inference by manipulating internal representations, aiming for faster and more lightweight output control (Turner et al., 2023; Turner et al., 2024; Wang et al., 2025b; Stolfo et al., 2025).

However, activation-level edits are often coarse: they mix multiple semantics, a phenomenon called polysemanticity (Bricken et al., 2023). Recently, Sparse Autoencoders (SAEs) have become a valuable tool in the interpretability field. They are trained to actively decompose the hidden states of the LLM into sparse and human readable features (Templeton et al., 2024; Mudide et al., 2025). Their interpretable nature has subsequently spurred research into leveraging SAE features for more precise, concept-level control over model behavior (Ferrando et al., 2025; Chalnev et al., 2024).

Despite this progress, a critical question remains unanswered: *does higher interpretability truly imply better utility?* Since SAEs are trained to balance reconstruction and sparsity to yield human-readable features (Cunningham et al., 2023; Makelov, 2024; O'Brien et al., 2025), their utility for downstream tasks is not a primary objective. Understanding and characterizing this gap is critical to enabling more interpretable and effective steering with SAEs, because (i) it clarifies when in-

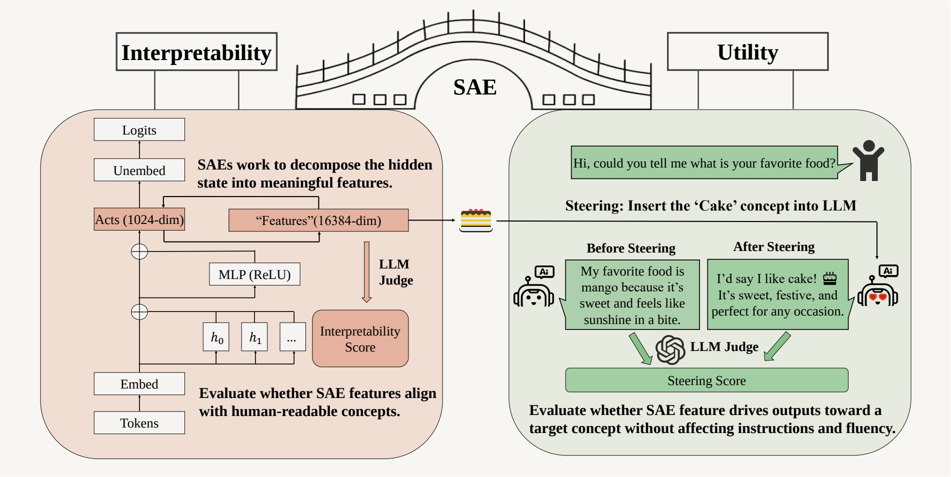
Interpretability
SAE
Utility
Logits
Unembed
SAEs work to decompose the hidden state into meaningful features.
Acts (1024-dim)
"Features"(16384-dim)
MLP (ReLU)
LLM Judge
h_0
h_1
...
Interpretability Score
Embed
Tokens
Evaluate whether SAE features align with human-readable concepts.
Hi, could you tell me what is your favorite food?
Steering: Insert the 'Cake' concept into LLM
Before Steering
After Steering
AI
My favorite food is mango because it's sweet and feels like sunshine in a bite.
I'd say I like cake! It's sweet, festive, and perfect for any occasion.
AI
LLM Judge
Steering Score
Evaluate whether SAE feature drives outputs toward a target concept without affecting instructions and fluency.

Figure 1: **Overview of our goal: building a bridge for SAE interpretability and utility. Interpretability (left):** an SAE attached to the LLM decomposes hidden states into sparse, human-describable features. An LLM judge yields an *interpretability score* for the SAE (Paulo et al., 2025). **Utility (right):** at inference, we activate a target SAE feature (e.g., 'cake') to steer generation. An LLM judge yields *steering utility score* (Wu et al., 2025).

terpretability scores can (and cannot) be used as a practical proxy for utility, and (ii) it motivates training objectives that explicitly balance reconstruction, interpretability, and downstream steering performance. To this end, we conduct a systematic study to build a bridge between SAE interpretability and steering utility (see Figure 1).

To perform a comprehensive association analysis, we train 90 SAEs across three LLMs (Gemma-2-2B (Team et al., 2024), Qwen-2.5-3B (Yang et al., 2024), and Gemma-2-9B) spanning diverse architectures and sparsity levels. We compute interpretability using SAEBench (Karvonen et al., 2025) and steering utility using AxBench (Wu et al., 2025). Then, we leverage a pairwise-controlled framework to evaluate whether interpretability predicts steering performance across the pool of trained SAEs. To quantify this relationship, we follow the idea of prior works (Jiang et al., 2020; Hu et al., 2024) and measure rank agreement between interpretability and utility using Kendall's rank coefficient τ_b. We control confounders with an axis-conditioned analysis, isolating each design axis (architecture, sparsity, model) by varying one at a time and aggregating per-axis metrics.

Furthermore, as identified in Arad et al. (2025); Wu et al. (2025), not all interpretable features in SAE are equally effective for steering. This motivates our next objective to identify the specific features critical for behavior control and steering utility analysis. Motivated by the recent progress on the entropy mechanism in LLM reasoning (Fu et al., 2025; Wang et al., 2025a), we propose an innovative selection criterion for SAE features: Δ *Token Confidence*, which measures the degree to which amplifying a single feature shifts the model's next-token distribution. Features that induce the most substantial change in model confidence are identified as high-utility candidates features for steering, as they exert a measurable and targeted influence on model behavior. Finally, we leverage these critical features to conduct a refined analysis of the interpretability-utility gap.

The primary contributions and insights of this paper are summarized as follows:

1. **(§3.4) Interpretability shows a relatively weak positive association with utility.** Across 90 SAEs that are trained across three model sizes, five architectures, and six sparsity levels, we find that a higher *interpretability score* tends to shows a relatively weak positive association with steering performance (the Kendall's rank coefficient $\tau_b \approx 0.298$)). This identifies a notable interpretability-utility gap of the existing SAEs.
2. **(§4.2) Δ *Token Confidence* effectively selects features with strong steering performance.** To identify the SAE features that are critical for steering, we introduce Δ *Token Confidence*, an innovative metric that identifies steering-critical SAE features by measuring their impact on the model's next-token distribution. When benchmarked against the best existing output score-based method (Arad et al., 2025), our approach yields a substantial 52.52% average improvement in

steering score. This result validates the superiority of our method and underscores the critical role of feature selection in characterizing and enhancing the steering utility of SAEs.

3. **(§4.3) The interpretability-utility gap widens among high-utility features.** By reapplying our association analysis exclusively to SAE features with strong steering utility, we uncover a counterintuitive finding: the interpretability-utility correlation vanishes or even becomes negative (Kendall's rank coefficient $\tau_b \approx 0$). This indicates that for the most effective steering features, interpretability is at best irrelevant and potentially detrimental, further emphasizing the critical nature of the interpretability-utility gap.

Our results demonstrate a significant divergence between SAE's interpretability and steering utility, suggesting that prioritizing interpretability does not enable improved steering performance. This gap highlights a crucial research direction: mitigating it will likely necessitate advanced post-training feature selection protocols or fundamentally new, utility-oriented SAE training paradigms.

2 Preliminary

2.1 Sparse Autoencoders

Sparse Autoencoders (SAEs) decompose internal model activations x into sparse, higher-dimensional features h that can be linearly decoded back to the original space (Cunningham et al., 2023; Leask et al., 2025). A standard SAE with column-normalized decoder weights (Bricken et al., 2023; Karvonen et al., 2024) is defined by the following forward map and optimization objective:

$$\mathcal{L} \;=\; \|x-\hat{x}\|_2^2 \;+\; \lambda\|h\|_1, \text{ where } h=\mathrm{ReLU}(W_E x+b_E),\; \hat{x}=W_D h+b_D,$$

where W_E, b_E are encoder parameters, W_D, b_D are decoder parameters, $\hat{x}$ is the reconstruction, and λ controls sparsity. This training balances reconstruction accuracy with sparse representations.

2.2 Interpretability: Automated Interpretability Score

SAEBench (Karvonen et al., 2025) uses an LLM-as-judge (Paulo et al., 2025) to assess each latent: the judge drafts the description from examples and then predicts, on a held-out set, which sequences activate it. The *Automated Interpretability Score* is the average precision of the judge's prediction.

$$\text{AutoInterp Score} \;=\; \frac{1}{M}\sum_{m=1}^{M}\mathbf{1}[\hat{y}_m = y_m]\,,$$

where $y_m \in \{0,1\}$ indicates whether the latent activates in the sequence m and $\hat{y}_m$ is the judge's prediction. We use this score as our interpretability metric. For the complete details, see Appendix B.

2.3 Utility: Steering Score

SAE steering injects the SAE decoder atom v_f (the f-th column of the column-normalized decoder $W_{\text{dec}}[f]$) into the residual stream at a target layer to push the hidden state x along a chosen feature direction (Durmus et al., 2024). Given a feature index f, a steering factor α, and a per-sample scale m_f (e.g., the feature's maximum activation), the intervention is

$$x^{\text{steer}} \;=\; x \;+\; (\alpha\, m_f)\cdot v_f. \tag{1}$$

Through the above formula (1), we can use SAE features for steering to achieve the output of controlling LLM. AxBench (Wu et al., 2025) measures causal control by steering internal representations during generation and asking an LLM judge to rate three aspects, each on a discrete scale $\{0,1,2\}$: *Concept* (C), *Instruction* (I), and *Fluency* (F). The overall *Steering Score* is the harmonic mean:

$$\text{Steering Score} \;=\; \mathrm{HM}(C,I,F) \;=\; \frac{3}{\frac{1}{C}+\frac{1}{I}+\frac{1}{F}} \;\in\; [0,2].$$

Following AxBench, for each concept we sample instructions (e.g., 10 from Alpaca-Eval (Dubois et al., 2023)), generate continuations under different steering factors, pick the best factor on one split, and evaluate the held-out split with the judge to obtain the final utility score averaged across prompts (Gu et al., 2025). The complete scoring procedure is detailed in Appendix C.

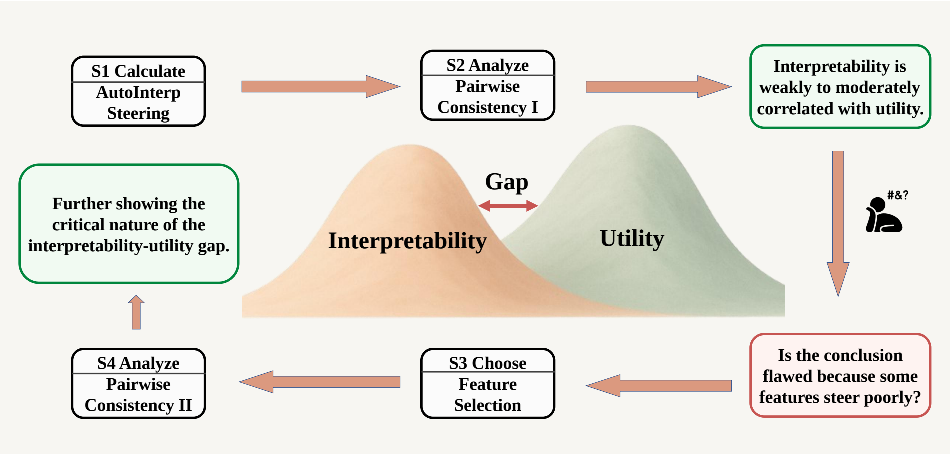
S1 Calculate
AutoInterp
Steering
S2 Analyze
Pairwise
Consistency I
Interpretability is weakly to moderately correlated with utility.
Gap
Interpretability
Utility
#&?
Further showing the critical nature of the interpretability-utility gap.
Is the conclusion flawed because some features steer poorly?
S4 Analyze
Pairwise
Consistency II
S3 Choose
Feature
Selection

Figure 2: **Overview of our pairwise-controlled workflow linking SAE interpretability with steering utility.** **(S1)** Compute *interpretability score* and *steering score* for each SAE. **(S2)** Pairwise analysis across SAEs and get an insight (the top-right green box), revealing an interpretability–utility gap. The red box (lower right) is our further inference based on the above green box and previous studies (Wu et al., 2025). **(S3)** Use Δ *Token Confidence* to select higher-utility features. **(S4)** Compute *steering score* after selection per SAE, then do the pairwise analysis between *steering score* and *interpretability*. The green box in the middle left is our final conclusion.

3 Can SAE Interpretability Indicate Steering Performance?

3.1 Experimental Setup

Dataset. For each trained SAE, we score 1,000 latents with *LLM-as-judge* (Paulo et al., 2025) and randomly sample 100 to form that SAE's Concept100: 100 human–readable concept descriptions per evaluation set, each pairing a latent (layer, feature_id) with a short natural–language description. (see details in Appendix F). For steering, we sample 10 Alpaca-Eval instructions, allow up to 128 generated tokens, and test 6 steering factors; the 10 instructions are split 5/5 for factor selection vs. held-out evaluation.

Model. We evaluate three open LLMs: Gemma-2-2B (Team et al., 2024), Qwen-2.5-3B (Yang et al., 2024), and Gemma-2-9B (Team et al., 2024). SAEs are trained on residual-stream activations at a fixed mid-layer for each model: Layer 12 for Gemma-2-2B, Layer 17 for Qwen-2.5-3B, and Layer 20 for Gemma-2-9B—and steering is applied to the corresponding layer.

SAE with different architectures We train 90 SAEs covering a range of architectures and sparsity. All SAEs use a latent dictionary width of 16k. We instantiate five variants: BatchTopK (Bussmann et al., 2024), Gated (Rajamanoharan et al., 2024a), JumpReLU (Rajamanoharan et al., 2024b), ReLU (Team, 2024), TopK (Gao et al., 2024) and sweep six target sparsity levels with approximate per-token activations $L_0 \approx 50, 80, 160, 320, 520, 820$. Further details are provided in Appendix A.

3.2 Pairwise Rank Consistency between Interpretability and Utility

We test whether higher interpretability of SAE is predictive of higher steering performance across a set of trained SAEs attached to a fixed LM. For each SAE θ in a pool Θ, we record a pair $(\mu(\theta),\ g(\theta)) \in \mathbb{R}^2$, where μ is the SAE-level *Interpretability Score* and g is an aggregated *Steering Score* over a standardized evaluation suite.

Given two SAEs $\theta_i, \theta_j \in \Theta$, define the concordance indicator

$$v_{ij} \;=\; \operatorname{sign}\big(\mu(\theta_i) - \mu(\theta_j)\big) \cdot \operatorname{sign}\big(g(\theta_i) - g(\theta_j)\big) \in \{-1, 0, +1\}. \tag{2}$$

Kendall's tie-corrected rank coefficient τ_b (Kendall, 1938) summarizes agreement over unordered pairs and reduces to average concordance when there are no ties:

$$\tau_b \;=\; \frac{\displaystyle\sum_{i<j} v_{ij}}{\sqrt{\left(\displaystyle\sum_{i<j} \mathbf{1}\left[\mu(\theta_i) \neq \mu(\theta_j)\right]\right)\left(\displaystyle\sum_{i<j} \mathbf{1}\left[g(\theta_i) \neq g(\theta_j)\right]\right)}} \in [-1, 1], \tag{3}$$

In this study, we instantiate μ with the *Interpretability Score* and g with the *Steering Score*, then compute τ for three model–layer settings (Gemma-2-2B, Qwen-2.5-3B, Gemma-2-9B). Each setting includes 30 SAEs spanning architectures and sparsity to ensure sufficient pair coverage.

3.3 Granulated Kendall's Coefficient to Control Confounders

Global rank agreement can be confounded by hyperparameters that jointly influence interpretability and utility. To obtain an axis-controlled assessment, we factor the SAE design space into orthogonal axes and evaluate rank consistency while varying one axis at a time and holding the others fixed.

We define three conditioning axes: (A) Architecture — fix architecture (and layer), vary sparsity; (B) Sparsity — compare architectures at matched sparsity ranks; (C) Model — fix the base model, compare all SAEs within it. For axis i, partition Θ into groups $\mathcal{G}_i$ that are matched on all axes except i. Within each group $G \in \mathcal{G}_i$, compute Kendall's coefficient in $\{(\mu(\theta), g(\theta)) : \theta \in G\}$, and average between groups to obtain the statistic at the axis level:

$$\psi_i \;=\; \frac{1}{|\mathcal{G}_i|} \sum_{G \in \mathcal{G}_i} \tau(\{(\mu(\theta), g(\theta)) : \theta \in G\})\,. \tag{4}$$

Aggregate the axis-level outcomes by

$$\Psi \;=\; \frac{1}{n} \sum_{i=1}^{n} \psi_i, \tag{5}$$

where n is the number of axes. Each ψ_i captures rank consistency conditioned on axis i (varying only that axis while matching the others), and Ψ aggregates these into a single axis-controlled measure. This construction mitigates cross-axis trends—e.g., architecture, sparsity, or model-driven shifts that can obscure local relationships between interpretability and utility.

We report the per-axis statistics ψ_i together with the aggregate Ψ for the same model settings as in section 3.2, providing both axis-specific and aggregated assessments.

3.4 Pairwise Analysis Results

In this section, we assess whether higher SAE interpretability predicts stronger steering by computing Kendall's τ_b between the *Interpretability Score* $\mu(\theta)$ and the *Steering Score* $g_{\text{base}}(\theta)$ (before any feature selection) over a pooled set of SAEs attached to a fixed LLM.

To control confounders and localize effects, we apply the axis-conditioned procedure defined in section 3.3. For each axis, we form matched groups, compute within-group τ_b, average to obtain a per-axis summary, and aggregate these summaries into an overall axis-controlled coefficient.

Table 1 shows that **across SAEs, higher interpretability tends to be modestly associated with better steering on average, pointing to a consistent but limited impact.** The pooled Kendall's $\tau_b \approx 0.30$ is positive, and the axis-controlled aggregate remains positive ($\Psi \approx 0.25$), indicating that more interpretable features generally translate into better steering utility across designs and models.

The strength of the link between interpretability and utility depends on SAE architecture, sparsity, and the base model. By architecture, the association is positive on average ($\Psi_{\text{A}} \approx 0.26$), with ReLU-like variants reinforcing the trend and Gated weakening it. By sparsity, alignment is strongest when the SAE is more sparse and weakens—sometimes reversing—as the number of active features increases. By model, the underlying LM shapes the effect, with the signal clearest in Qwen-2.5-3B and weaker in Gemma-2-2B, while the model-wise summary remains positive ($\Psi_{\text{C}} \approx 0.33$).

Table 1: **Pairwise Analysis Between *Interpretability Score* $\mu(\theta)$ and *Steering* Score $g_{\text{base}}(\theta)$.** Here $g_{\text{base}}(\theta)$ is steering score before any feature selection. We report Kendall's τ_b overall and by axis-controlled measures Ψ_{A} (Architecture), Ψ_{B} (Matched Sparsity), and Ψ_{C} (Model). n = number of SAEs; Pairs = number of pairwise comparisons; p = permutation p-value for testing $H_0 : \tau_b = 0$ within each group (by shuffling steering scores); 95% CI = 95% bootstrap percentile confidence interval for τ_b.

Axis	**SAEs**	n	**Pairs**	τ_b	p	**95% CI**
Overall	All SAEs	90	4005	**0.2979**	0.0002	[0.1591, 0.4191]
$\Psi_{\mathrm{A}} =$ **0.2575** (SE $\approx$ 0.1163, 95% boot CI [0.0222, 0.3961])						
Ψ_{A}: Architecture	BatchTopK	18	153	0.3203	0.0712	[0.0000, 0.6429]
	Gated	18	153	-0.2026	0.2577	[−0.5493, 0.1918]
	JumpReLU	18	153	0.4248	0.0160	[0.0563, 0.7183]
	ReLU	18	153	0.3595	0.0392	[0.0274, 0.6528]
	TopK	18	153	0.3856	0.0272	[0.0282, 0.6966]
$\Psi_{\mathrm{B}} =$ **0.1651** (SE $\approx$ 0.1112, 95% boot CI [−0.0286, 0.3587])						
Ψ_{B}: Sparsity	$L_0 \approx 50$	15	105	0.5429	0.0034	[0.1578, 0.8737]
	$L_0 \approx 80$	15	105	0.3524	0.0740	[0.0000, 0.6304]
	$L_0 \approx 160$	15	105	0.1810	0.3821	[−0.2501, 0.5464]
	$L_0 \approx 320$	15	105	0.1810	0.3673	[−0.1579, 0.5208]
	$L_0 \approx 520$	15	105	-0.2190	0.2837	[−0.5914, 0.1648]
	$L_0 \approx 820$	15	105	-0.0476	0.8484	[−0.4409, 0.3750]
$\Psi_{\mathrm{C}} =$ **0.3272** (SE $\approx$ 0.0698, 95% boot CI [0.2184, 0.4575])						
Ψ_{C}: Model	Gemma-2-2B	30	435	0.2184	0.0980	[−0.0644, 0.4764]
	Qwen-2.5-3B	30	435	0.4575	0.0008	[0.2086, 0.6580]
	Gemma-2-9B	30	435	0.3057	0.0166	[0.0521, 0.5277]
$\Psi = \big(\Psi_{\mathrm{A}} + \Psi_{\mathrm{B}} + \Psi_{\mathrm{C}}\big)/3 \; = \;$ **0.2499**						

Key Observation 1: Interpretability shows a relatively weak positive correlation with steering performance, highlighting a notable gap between interpretability and utility across SAEs.

4 From Interpretability to Utility: Which SAE Features Actually Steer?

In Sec. 3.4, We find that SAE interpretability is a relatively weak prior for steering utility. Prior work (Arad et al., 2025) shows many features lack steerability and we speculate that this factor may render the previous conclusion inaccurate. Therefore, we introduce a metric to identify steering-effective features. Metrics derived from a model's internal token distributions can assess reasoning quality (Kang et al., 2025). In particular, *token entropy* offers a unified view: high entropy highlights critical decision points (Fu et al., 2025; Wang et al., 2025a). We apply this idea to SAE steering.

4.1 Feature Selection via Δ Token Confidence

We start from the model's next-token distribution. Given logits $z \in \mathbb{R}^V$ and $p = \mathrm{softmax}(z)$ over a vocabulary of size V, the *token entropy* is

$$H(p) \; = \; -\sum_{j=1}^{V} p_j \, \log p_j, \tag{6}$$

Entropy summarizes dispersion over the vocabulary: smaller values reflect a sharper, more concentrated prediction, while larger values indicate greater uncertainty at a given position.

To focus on the head of the distribution that matters most for sampling, we use *token confidence*. Let $\mathcal{I}_k(p) \subseteq \{1, \ldots, V\}$ denote the indices of the k largest probabilities in p. The top-k *token confidence* is the negative average log-probability over these entries:

$$C_k(p) \; = \; -\frac{1}{k} \sum_{j \in \mathcal{I}_k(p)} \log p_j. \tag{7}$$

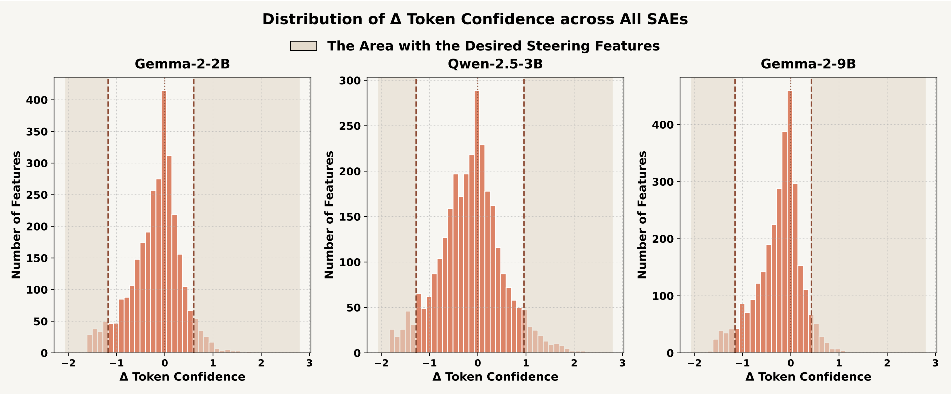

Figure 3: **Distribution of per-feature Δ *Token Confidence* across all SAEs.** Panels show histograms for Gemma-2-2B, Qwen-2.5-3B, and Gemma-2-9B; the x-axis is ΔC_k (negative values indicate increased confidence, positive values decreased confidence) and the y-axis is the number of SAE features. The shaded area marks the high-magnitude tails from which candidate steering features are selected, while the central mass near 0 indicates features with little distributional impact.

Lower C_k implies higher confidence, while higher C_k implies a flatter top-k distribution. Unlike entropy, C_k directly captures the sharpness of the outcomes that drive next-token behavior.

We turn confidence into a feature-level selector via a single-feature SAE intervention. Consider an SAE feature f at layer ℓ. We amplify only the coefficient of f by a factor $\alpha > 0$ in the SAE reconstruction, leaving the base model and all other features unchanged. Denote the baseline next-token distribution by p^{base} and the intervened distribution by $p^{\text{int}}_{f,\ell,\alpha}$. Δ *Token Confidence* is

$$\Delta C_k(f;\ell,\alpha) \;=\; C_k\big(p^{\text{int}}_{f,\ell,\alpha}\big) \;-\; C_k\big(p^{\text{base}}\big). \tag{8}$$

Negative values $\Delta C_k < 0$ mean that amplifying f sharpens the top-k distribution, while positive values indicate greater dispersion. We compute this using one baseline and one intervened forward pass via an SAE hook. Implementation details and hyperparameters are provided in Appendix D.

We select features with extreme signed changes in *token confidence* under single-feature SAE interventions, and pick the better direction (see Figure 3). For each feature, we compute ΔC_k, rank by $|\Delta C_k|$, form tiers, evaluate subsets for steering, and keep the best per SAE.

4.2 Steering Performance Results After Feature Selection

Arad et al. (2025) has shown that SAE steering works well if features are chosen by their causal impact on model outputs, introducing the *output score* as a metric to identify output-aligned features. For a given SAE feature, a logit-lens procedure is used to select a representative token set M. Let $P_{\text{base}}(M)$ and $P_{\text{int}}(M)$ denote the aggregate support for M in the base and intervened (feature-amplified) runs, respectively, where

$$P(M) = \left(1 - \frac{\min_{i\in M}\operatorname{rank}(i)}{|V|}\right) \max_{i\in M} p(i),$$

with $|V|$ the vocabulary size, $\operatorname{rank}(i)$ the rank of token i, and $p(i)$ its probability. The single-feature steering score is then

$$S_{\text{out}} = P_{\text{int}}(M) - P_{\text{base}}(M),$$

which is large when amplifying the feature increases the rank and probability of representative token set. Following this insight, we evaluate our Δ *token confidence* selection on three base LLMs (Gemma-2-2B, Qwen-2.5-3B, Gemma-2-9B) using the CONCEPT100 (see details in 3.1). We set $k = 1$ in our Δ *token confidence* selection. In ablations over $k \in \{1, 3, 5, 10\}$ on Gemma and Qwen, all choices of k yield large gains over the SAE baseline, and $k = 1$ is in fact the best-performing setting on both models, so we adopt it as our default (see Appendix L). The experiments on steering performance improvement of each SAE can be referred to Appendix E.

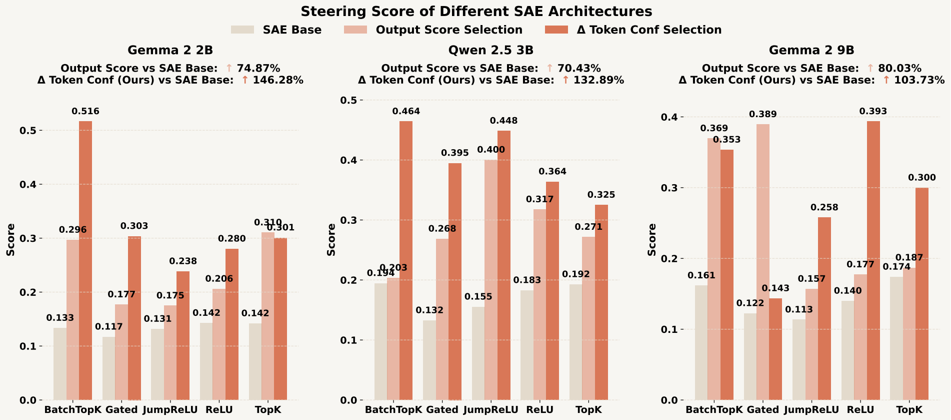

Figure 4: **Comparison of different SAE steering methods with five SAE architecture across three LLMs.** Panels correspond to Gemma-2-2B, Qwen-2.5-3B, and Gemma-2-9B. The horizontal axis groups SAE architectures (BatchTopK, Gated, JumpReLU, ReLU, TopK), and the vertical axis reports the *steering score*. Bars show three conditions: SAE Base (no feature selection), Output Score Selection, and Δ *Token Confidence* Selection (ours). Panel annotations summarize the average lift of each selection method relative to the SAE-based steering.

Table 2: ***Steering score* after feature selection compared with SAE-based steering.** Columns report scores (higher is better) for Gemma-2-2B, Qwen-2.5-3B, and Gemma-2-9B. Rows: 'SAE-based' uses all SAE features without selection (Wu et al., 2025); '+Output' selects features using S_{out} (Arad et al., 2025); "+ΔC_k (Ours)" selects by the Δ *Token Confidence*. Boldface indicates the best method per model.

Method	**Gemma-2-2B**	**Qwen-2.5-3B**	**Gemma-2-9B**
SAE-based	0.133	0.171	0.142
+Output	0.233	0.292	0.255
+ΔC_k(Ours)	**0.328**	**0.399**	**0.289**

Table 2 shows that our selection yields consistent gains across all models, outperforming the vanilla SAE baseline by large margins, and also improving over an output-score–based selector. These gains indicate that ranking and filtering by the magnitude of distributional change captured by ΔC_k reliably isolates features with the strongest steering utility.

Furthermore, we conducted a comparative analysis of SAEs of different architectures on three models. For fair comparison, the two feature selection methods use the same subset size. Figure 4 compares *steering scores* across SAE architectures and selection methods. In all three models, selecting features by Δ *Token Confidence* consistently outperforms both the no-selection SAE baseline and the output-score selector across architectures.

On average, our method improves steering performance by **52.52%** over the strongest competing baseline. The BatchTopK architecture is the one that has the most stable and significant improvement in steering capabilities on models of different sizes among the five SAE architectures.

Key Observation 2: Δ *Token Confidence* reliably selects high-utility SAE features across models. Among SAE architectures, BatchTopK achieves the most stable and sizable *steering score*.

4.3 Pairwise Analysis After High-Confidence Feature Selection

Building on the high Δ *Token Confidence* feature selection introduced above, we now ask whether SAE interpretability can serve as a prior for *post-selection steering performance*. For each SAE θ, we write $g_{\text{base}}(\theta)$ for its *Steering Score* before feature selection and $g_{\text{high}}(\theta)$ for the best score achieved after selecting features with high Δ *Token Confidence*. Both quantities are measured in the same steering metric; $g_{\text{high}}(\theta)$ is simply the steering score recomputed on a restricted set of high Δ *Token Confidence* features for the same SAE.

Table 3: **Pairwise Analysis Between *Interpretability Score* $\mu(\theta)$ and *Steering Score* $g_{\textbf{high}}(\theta)$.** Here $g_{\text{high}}(\theta)$ is steering score after selecting features with high Δ Token Confidence. We report Kendall's τ_b overall and under axis-controlled summaries Ψ_{A} (Architecture), Ψ_{B} (Matched Sparsity), and Ψ_{C} (Model). n = number of SAEs; Pairs = number of pairwise comparisons; p = permutation p-value for testing $H_0 : \tau_b = 0$ within each group (by shuffling steering scores); 95% CI = 95% bootstrap percentile confidence interval for τ_b.

Axis	**SAEs**	n	**Pairs**	τ_b	p	**95% CI**
Overall	All SAEs	90	4005	**0.0823**	0.2645	$[-0.0560, 0.2243]$
$\Psi_{\text{A}} =$ **0.0618** (SE ≈ 0.1136, 95% boot CI $[-0.1149, 0.2810]$)						
$\Psi_{\textbf{A}}$**: Architecture**	BatchTopK	18	153	-0.1895	0.2933	$[-0.5694, 0.1918]$
	Gated	18	153	0.0338	0.8764	$[-0.3656, 0.3853]$
	JumpReLU	18	153	0.0950	0.6209	$[-0.3373, 0.4697]$
	ReLU	18	153	-0.1007	0.5803	$[-0.4833, 0.2857]$
	TopK	18	153	0.4702	0.0078	$[0.0794, 0.8043]$
$\Psi_{\text{B}} =$ **0.1002** (SE ≈ 0.0700, 95% boot CI $[-0.0288, 0.2188]$)						
$\Psi_{\textbf{B}}$**: Sparsity**	$L_0 \approx 50$	15	105	0.2488	0.2212	$[-0.1828, 0.6300]$
	$L_0 \approx 80$	15	105	0.2899	0.1602	$[-0.0673, 0.5876]$
	$L_0 \approx 160$	15	105	0.1183	0.5747	$[-0.2357, 0.4151]$
	$L_0 \approx 320$	15	105	0.1171	0.5803	$[-0.3531, 0.5209]$
	$L_0 \approx 520$	15	105	-0.1827	0.3747	$[-0.6490, 0.3524]$
	$L_0 \approx 820$	15	105	0.0097	1.0000	$[-0.3724, 0.3572]$
$\Psi_{\text{C}} =$ **0.0538** (SE ≈ 0.0700, 95% boot CI $[-0.0540, 0.1850]$)						
$\Psi_{\textbf{C}}$**: Model**	Gemma-2-2B	30	435	-0.0540	0.6963	$[-0.3127, 0.2125]$
	Qwen-2.5-3B	30	435	0.1850	0.1668	$[-0.0872, 0.4413]$
	Gemma-2-9B	30	435	0.0303	0.8284	$[-0.2552, 0.3233]$
$\Psi = \big(\Psi_{\text{A}} + \Psi_{\text{B}} + \Psi_{\text{C}}\big)/3 =$ **0.0719**						

We quantify the relationship between interpretability and post-selection steering by computing Kendall's τ_b between the *Interpretability Score* $\mu(\theta)$ of SAEs and the *Steering* Score $g_{\text{high}}(\theta)$. As in Section 3.4, we report both pooled coefficients and axis-conditioned summaries that control for design and model factors (architecture, sparsity and model). In addition, Section 3.4 and Table 1 report the corresponding coefficients for $\mu(\theta)$ vs. $g_{\text{base}}(\theta)$, allowing a direct comparison between "before" and "after" feature selection using the same steering metric.

Overall, Table 3 shows that interpretability is not a reliable prior for steering performance after selection. The pooled association between $\mu(\theta)$ and $g_{\text{high}}(\theta)$ is small and statistically indistinguishable from zero ($\tau_b \approx 0.08$), and the axis-controlled aggregate is likewise close to zero ($\Psi \approx 0.07$). Estimates across architectures, sparsity levels, and models cluster around zero and are mostly non-significant. These results indicate that once we focus on features that are most useful for steering (as selected by high Δ *Token Confidence*), higher interpretability can't predict better steering scores.

Key Observation 3: Surprisingly, the interpretability–utility gap widens after feature selection: higher interpretability scores can't provide a prior for which SAEs achieve better steering.

5 Related Work

5.1 Representation-Based Steering

Activation-based steering arose as a lightweight alternative to fine-tuning, enabling on-the-fly control of LLM behavior without retraining (Giulianelli et al., 2018; Vig et al., 2020; Geiger et al., 2021; 2025). The core idea is to inject carefully chosen directions into hidden states, typically in the residual stream, scaling interventions by a gain and selecting layers for maximal effect (Zou et al., 2025; Rimsky et al., 2024; van der Weij et al., 2024). It has been applied to safety and moderation, persona and sentiment control, and instruction adherence, promising low-latency deployment-time adjustment but facing polysemantic entanglement and brittleness that motivate standardized evaluation (Chen et al., 2025; Liu et al., 2024). However, this approach injects polysemantic activations at intervention time, yielding coarse-grained effects for output control (Bricken et al., 2023). Our work is related to activation-level interventions, but differs by grounding directions in sparse, interpretable SAE features and applying utility-oriented feature selection to mitigate these failure modes.

5.2 SAE-Based Steering

Sparse Autoencoders (SAEs) decompose activations into sparse, human-readable features to mitigate polysemanticity and expose concept-level structure (Bricken et al., 2023; Templeton et al., 2024; Gao et al., 2024). For steering, practitioners use decoder atoms as directions and add scaled injections at chosen layers, with architecture and sparsity choices trading reconstruction for feature granularity (Zhao et al., 2025; Wang et al., 2025d; Ferrando et al., 2025). SAE-based steering enables targeted safety control, style modulation, and instruction emphasis, yet the utility of individual features varies widely (Chalnev et al., 2024; Mayne et al., 2024). While the connection between SAE interpretability and steering utility remains unclear, and our goal is to build a principled bridge between them. To this end, we conduct a large-scale experiments across multiple model sizes and SAE architectures, demonstrating the critical nature of the interpretability-utility gap.

6 Conclusion and Discussion

In summary, SAE interpretability shows relatively weak positive association with steering utility across 90 SAEs ($\tau_b \approx 0.298$), revealing a clear interpretability–utility gap. Selecting features with Δ *Token Confidence* yields substantial gains (average $+52.52\%$ over the strongest existing baseline). Surprisingly, when analyzing *steering score* after selection, the correlation with interpretability collapses toward zero and can even turn negative for the highest-utility features, further underscoring this gap. This gap points to a key direction: develop task-general utility indicators that reliably predict steerability across models, or design training objectives that directly optimize controllability under sparsity so features are utility-calibrated without heavy post-hoc selection. Our work provides valuable insight for the further development of SAEs as interpretable tools.

Reproducibility Statement

We aim to facilitate full reproduction of our results. All model code, training and evaluation scripts, configuration files, and experiment logs are released at an *anonymous* repository as part of the supplementary materials: https://anonymous.4open.science/r/SAE4Steer. Training architectures, hyperparameters, sparsity schedules, and optimization details are specified in the main text and Appendix A.2 (see also the per-family settings in Appendix A). The datasets used are openly licensed: all SAEs are trained on The Common Pile v0.1 (Kandpal et al., 2025) as described in Appendix F; our evaluation concepts (Concept100) and their automatic generation pipeline are documented in Appendix B and Appendix F. The complete procedures for automated interpretability scoring (SAEBench) and steering utility (AxBench), including sampling, judging protocols, and scoring functions, are detailed in Appendix B and Appendix C, with the Δ *Token Confidence* selector defined in Appendix D and the post-selection results summarized in Appendix E. Hardware, runtime, and memory footprints for both SAEBench and AxBench are reported in Appendices B.3 and C.2. Together, these materials, along with seed-controlled configuration files and exact command-line invocations provided in the anonymous repository, are intended to enable independent researchers to replicate and extend our findings.

References

Dana Arad, Aaron Mueller, and Yonatan Belinkov. Saes are good for steering – if you select the right features, 2025. URL `https://arxiv.org/abs/2505.20063`.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk: A simple improvement for topksaes. In *AI Alignment Forum*, pp. 17, 2024.

Sviatoslav Chalnev, Matthew Siu, and Arthur Conmy. Improving steering vectors by targeting sparse autoencoder features, 2024. URL https://arxiv.org/abs/2411.02193.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models, 2025. URL https://arxiv.org/abs/2507.21509.

Yihan Chen, Benfeng Xu, Quan Wang, Yi Liu, and Zhendong Mao. Benchmarking large language models on controllable generation under diversified instructions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17808–17816, 2024.

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

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpacafarm: a simulation framework for methods that learn from human feedback. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 30039–30069, 2023.

Esin Durmus, Alex Tamkin, Jack Clark, Jerry Wei, Jonathan Marcus, Joshua Batson, Kunal Handa, Liane Lovitt, Meg Tong, Miles McCain, Oliver Rausch, Saffron Huang, Sam Bowman, Stuart Ritchie, Tom Henighan, and Deep Ganguli. Evaluating feature steering: A case study in mitigating social biases, 2024. URL https://anthropic.com/research/evaluating-feature-steering.

Javier Ferrando, Oscar Balcells Obeso, Senthooran Rajamanoharan, and Neel Nanda. Do i know this entity? knowledge awareness and hallucinations in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WCRQFlji2q.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence, 2025. URL https://arxiv.org/abs/2508.15260.

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, 2024. URL https://arxiv.org/abs/2406.04093.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 9574–9586. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/4f5c422f4d49a5a807eda27434231040-Paper.pdf.

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025. URL http://jmlr.org/papers/v26/23-0058.html.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 240–248, 2018.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on llm-as-a-judge, 2025. URL https://arxiv.org/abs/2411.15594.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yunzhe Hu, Difan Zou, and Dong Xu. An in-depth investigation of sparse rate reduction in transformer-like models. *Advances in Neural Information Processing Systems*, 37:116815–116837, 2024.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: Llm-based input-output safeguard for human-ai conversations, 2023. URL https://arxiv.org/abs/2312.06674.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. In *NeurIPS*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/4dbb61cb68671edc4ca3712d70083b9f-Abstract-Datasets_and_Benchmarks.html.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgIPJBFvH.

Nikhil Kandpal, Brian Lester, Colin Raffel, Sebastian Majstorovic, Stella Biderman, Baber Abbasi, Luca Soldaini, Enrico Shippole, A. Feder Cooper, Aviya Skowron, Shayne Longpre, Lintang Sutawika, Alon Albalak, Zhenlin Xu, Guilherme Penedo, Loubna Ben Allal, Elie Bakouch, John David Pressman, Honglu Fan, Dashiell Stander, Guangyu Song, Aaron Gokaslan, John Kirchenbauer, Tom Goldstein, Brian R. Bartoldson, Bhavya Kailkhura, and Tyler Murray. The Common Pile v0.1: An 8TB Dataset of Public Domain and Openly Licensed Text. *arXiv preprint arXiv:2506.05209*, 2025.

Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable best-of-n selection for large language models via self-certainty, 2025. URL https://arxiv.org/abs/2502.18581.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 83091–83118. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/9736acf007760cc2b47948ae3cf06274-Paper-Conference.pdf.

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 *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=qrU3yNfX0d.

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30:81–93, 1938. URL https://api.semanticscholar.org/CorpusID:120478295.

Aounon Kumar, Chirag Agarwal, Suraj Srinivas, Aaron Jiaxun Li, Soheil Feizi, and Himabindu Lakkaraju. Certifying LLM safety against adversarial prompting. In *Proceedings of the Conference on Language Modeling (COLM)*, 2024. URL https://arxiv.org/abs/2309.02705.

Patrick Leask, Bart Bussmann, Michael T Pearce, Joseph Isaac Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. Sparse autoencoders do not find canonical units of analysis. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9ca9eHNrdH.

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32287–32307. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24bx.html.

Aleksandar Makelov. Sparse autoencoders match supervised features for model steering on the IOI task. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024. URL https://openreview.net/forum?id=JdrVuEQih5.

Harry Mayne, Yushi Yang, and Adam Mahdi. Can sparse autoencoders be used to decompose and interpret steering vectors?, 2024. URL https://arxiv.org/abs/2411.08790.

Anish Mudide, Joshua Engels, Eric J. Michaud, Max Tegmark, and Christian Schroeder de Witt. Efficient dictionary learning with switch sparse autoencoders, 2025. URL https://arxiv.org/abs/2410.08201.

Kyle O'Brien, David Majercak, Xavier Fernandes, Richard Edgar, Blake Bullwinkel, Jingya Chen, Harsha Nori, Dean Carignan, Eric Horvitz, and Forough Poursabzi-Sangdeh. Steering language model refusal with sparse autoencoders, 2025. URL https://arxiv.org/abs/2411.11296.

Gonçalo Santos Paulo, Alex Troy Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=EemtbhJOXc.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders, 2024a. URL https://arxiv.org/abs/2404.16014.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders, 2024b. URL https://arxiv.org/abs/2407.14435.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, 2024.

Alessandro Stolfo, Vidhisha Balachandran, Safoora Yousefi, Eric Horvitz, and Besmira Nushi. Improving instruction-following in language models through activation steering. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=wozhdnRCtw.

Anthropic Interpretability Team. Training sparse autoencoders. https://transformer-circuits.pub/2024/april-update/index.html#training-saes, 2024. Accessed: 2025-01-20.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van

Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

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

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering Language Models With Activation Engineering. *arXiv e-prints*, art. arXiv:2308.10248, August 2023. doi: 10.48550/arXiv.2308.10248.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2024. URL https://arxiv.org/abs/2308.10248.

Teun van der Weij, Massimo Poesio, and Nandi Schoots. Extending activation steering to broad skills and multiple behaviours, 2024. URL https://arxiv.org/abs/2403.05767.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning, 2025a. URL https://arxiv.org/abs/2506.01939.

Tianlong Wang, Xianfeng Jiao, Yinghao Zhu, Zhongzhi Chen, Yifan He, Xu Chu, Junyi Gao, Yasha Wang, and Liantao Ma. Adaptive activation steering: A tuning-free llm truthfulness improvement method for diverse hallucinations categories. In *Proceedings of the ACM on Web Conference 2025*, pp. 2562–2578, 2025b.

Xu Wang, Yan Hu, Wenyu Du, Reynold Cheng, Benyou Wang, and Difan Zou. Towards understanding fine-tuning mechanisms of LLMs via circuit analysis. In *Forty-second International*

Conference on Machine Learning, 2025c. URL `https://openreview.net/forum?id=45EIiFd6Oa`.

Xu Wang, Zihao Li, Benyou Wang, Yan Hu, and Difan Zou. Model unlearning via sparse autoencoder subspace guided projections. In *ICML 2025 Workshop on Machine Unlearning for Generative AI*, 2025d. URL `https://openreview.net/forum?id=MIlqM98o9I`.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D Manning, and Christopher Potts. Axbench: Steering LLMs? even simple baselines outperform sparse autoencoders. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=K2CckZjNy0`.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024. URL `https://api.semanticscholar.org/CorpusID:274859421`.

Yu Zhao, Alessio Devoto, Giwon Hong, Xiaotang Du, Aryo Pradipta Gema, Hongru Wang, Xuanli He, Kam-Fai Wong, and Pasquale Minervini. Steering knowledge selection behaviours in LLMs via SAE-based representation engineering. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5117–5136, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. URL `https://aclanthology.org/2025.naacl-long.264/`.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2025. URL `https://arxiv.org/abs/2310.01405`.

LLM Usage

In preparing this paper, large language models (LLMs) were used as an assistive tool for minor language polishing and stylistic improvements. All technical contributions, results, and conclusions are solely the work of the authors.

A SAE Architectures and Training Details

We train **90 SAEs** (30 per base model) across five architectures and six target sparsity levels. Unless stated otherwise, the dictionary width is **16K** codes (F=16,384), SAEs are attached to the residual stream at the layer described in the main text, and decoder columns are ℓ_2–normalized. All models are trained on The Common Pile v0.1 (Kandpal et al., 2025).

A.1 Architectures and Parameterization

We list the five SAE families with their named parameters (as implemented) and the corresponding shapes. The last column records architecture-specific thresholding/gating fields when present. Shapes assume residual dimension d=2304 and dictionary width F=16,384.

Architectures	W_{enc}	b_{enc}	W_{dec}	b_{dec}	**Threshold / Extras**
ReLU	encoder.weight: shape (16,384, 2,304)	encoder.bias: shape (16,384)	decoder.weight: shape (2,304, 16,384)	bias: shape (2,304)	—
Gated	encoder.weight: shape (16,384, 2,304)	gate_bias: shape (16,384)	decoder.weight: shape (2,304, 16,384)	decoder_bias: shape (2,304)	r_mag: shape (16,384); mag_bias: shape (16,384)
TopK	encoder.weight: shape (16,384, 2,304)	encoder.bias: shape (16,384)	decoder.weight: shape (2,304, 16,384)	b_dec: shape (2,304)	k
BatchTopK	encoder.weight: shape (16,384, 2,304)	encoder.bias: shape (16,384)	decoder.weight: shape (2,304, 16,384)	b_dec: shape (2,304)	k
JumpReLU	W_enc: shape (2,304, 16,384)	b_enc: shape (16,384)	W_dec: shape (16,384, 2,304)	b_dec: shape (2,304)	threshold: shape (16,384)

A.2 Training, Sparsity, and Compute Setup

Optimization and schedule. Adam with learning rate 3×10^{-4}; LR warmup 1000 steps; sparsity warmup 5000 steps; LR decay starting at 80% of total steps. Precision: `bfloat16`. LM batch size $= 4$, context length $= 2048$, SAE batch size $= 2048$. Each run trains on $\sim 5\times10^{8}$ tokens.

Sparsity controls. We sweep six target activity levels

$$L_0 \approx \{50,\ 80,\ 160,\ 320,\ 520,\ 820\}.$$

For TopK/BatchTopK we set k equal to the chosen L_0 (aux-k coefficient $1/32$; moving-threshold momentum 0.999; threshold tracking begins at step 1000). JumpReLU uses the same set via `target_l0`. For L_1–penalized families, we search the following penalty grids:

Family	L_1 **penalty values (used to span sparsity levels)**
Standard / Standard-New	0.012, 0.015, 0.020, 0.030, 0.040, 0.060
Gated SAE	0.012, 0.018, 0.024, 0.040, 0.060, 0.080

Training details. All training uses two NVIDIA RTX A800 GPUs. The table below reports the aggregated artifacts and training time (hours) for *30 SAEs per model* (total 90), together with the runtime configuration. Times and sizes are approximate.

Model	**#SAEs**	**Disk (GB)**	**Traing Time (H)**	**LM Batch**	**Context**	**SAE Batch**	**Peak Mem (GB)**
Gemma-2-2B	30	8.7	17	4	2048	2048	20
Gemma-2-9B	30	13.2	60	4	2048	2048	70
Qwen-2.5-3B	30	7.7	37	4	2048	2048	30

B SAEBench Details, Results and Our Costs

B.1 Automated Interpretability Score Process

SAEBench (Karvonen et al., 2025) follow an LLM-as-judge pipeline to assign an *automated interpretability score* to each SAE latent. First, we collect layer activations by running the base LM with caching and encoding the residual stream through the SAE to obtain $h \in \mathbb{R}^{N\times L\times F}$. We define a token window of length 21 (buffer = 10) around any center (i, t) and, unless stated otherwise, mask BOS/PAD/EOS positions. For a latent ℓ, we sample three window types: (i) **Top** ($n = 12$ non-overlapping peaks of $h[:, :, \ell]$), (ii) **Importance-Weighted** ($n = 7$, sampled proportional to activation after removing values at least as large as the smallest Top peak), and (iii) **Random** ($n = 10$, uniform over valid centers). Let $v_{\max}$ be the maximum activation seen in any Top window position and set a global threshold $\tau_{\text{act}} = 0.01\, v_{\max}$.

We split the sampled windows into a **generation set** (10 Top + 5 IW) and a **scoring set** (2 Top + 2 IW + 10 Random, shuffled). In generation, tokens with activation $> \tau_{\text{act}}$ are bracketed to highlight evidence; the judge LLM receives these 15 windows and returns a short English description of when the latent fires. In scoring, the judge sees the description and the 14 held-out windows *without* highlights and outputs a comma-separated list of indices it predicts as activations (or `None`).

Ground truth for a window $\mathcal{W}$ is $\mathbb{1}[\max_{u\in\mathcal{W}} h[u, \ell] > \tau_{\text{act}}]$. The per-latent score is the **accuracy** over the $M = 14$ scoring windows, i.e.,

$$\text{Score}(\ell) = \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}\big[\widehat{y}_m = y_m\big],$$

where $\widehat{y}_m \in \{0, 1\}$ is the judge prediction and y_m is the label defined above. For each SAE θ, we evaluate 1,000 latents and report the mean over a random Concept100 subset:

$$\mu(\theta) = \frac{1}{100}\sum_{\ell\in\text{Concept100}} \text{Score}(\ell).$$

B.2 Performance of SAEs on three models on SAEbench

Across the three backbones, the six SAEbench metrics (for information about these indicators, see SAEBench (Karvonen et al., 2025))jointly reveal how sparsity mechanisms balance interpretability, faithfulness, and causal structure. Automated Interpretability is strongest when encoders enforce compact latent usage (e.g., TopK/BatchTopK and ReLU at lower L_0), and it gradually softens as capacity expands. The Absorption metric (considered via its complement in the plots) indicates that designs concentrating signal into a small set of latents are less prone to feature stealing, whereas higher effective capacity encourages redundancy and competition across latents. Meanwhile, Core/Loss-Recovered remains uniformly high, showing that even sparse codes closely preserve original model behavior; increasing L_0 pushes faithfulness toward a ceiling without overturning the core trade-offs visible in the other metrics.

Gemma-2-2B. As shown in Fig. 5, Gemma-2-2B exhibits a balanced profile: interpretability stays robust for TopK/BatchTopK and ReLU at modest sparsity; absorption is contained when the code remains compact; and Core is near-saturated across the range. Improvements in SCR@20 are steady but measured, suggesting targeted debiasing with small k. Sparse Probing indicates that relatively few latents already carry much of the predictive signal, while RAVEL strengthens with moderate capacity, reflecting cleaner separation of attributes without undermining compactness.

Qwen-2.5-3B. For Qwen-2.5-3B (Fig. 6), interpretability at low–to–moderate L_0 is competitive—especially for TopK and JumpReLU—yet the model is more sensitive to absorption as capacity grows, implying greater latent competition and signal spread. Core remains excellent, so reconstructions are faithful; however, SCR gains can flatten at high L_0 where residual spurious cues reappear. Sparse Probing is solid but a touch behind the strongest Gemma configurations, consistent with its flatter RAVEL patterns: causal structure is present but less crisply disentangled when attributes begin to diffuse across latents.

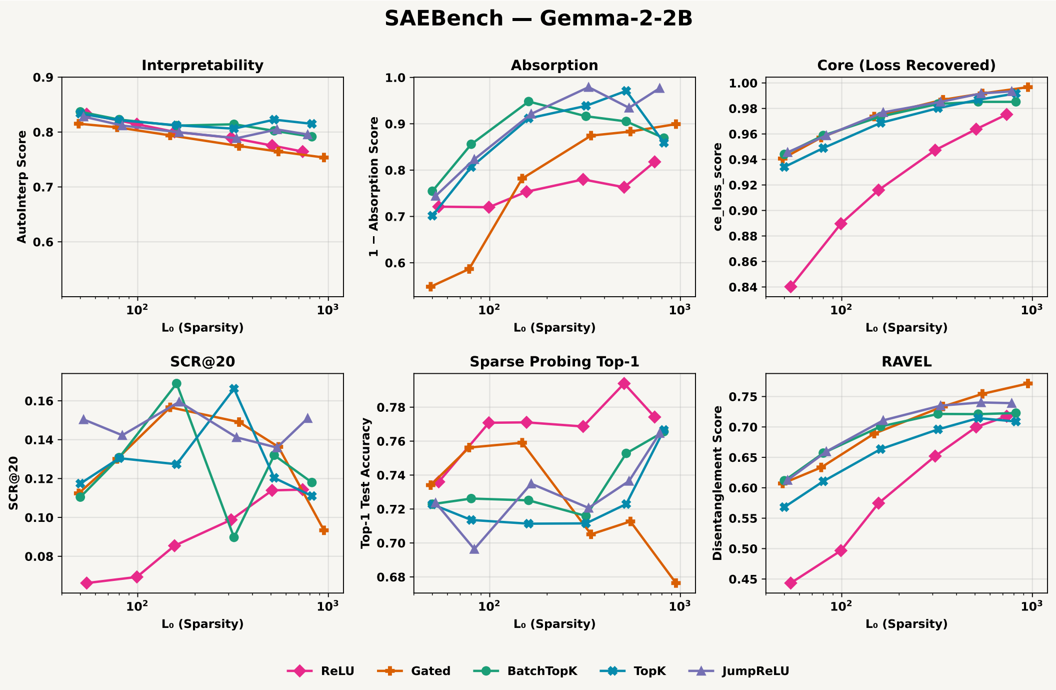

Figure 5: SAEbench results for **Gemma-2-2B**: interpretability remains strong at lower L_0, absorption stays low for compact codes, Core is near ceiling, and structure (SCR/RAVEL) improves with moderate capacity.

Gemma-2-9B. Gemma-2-9B (Fig. 7) pushes the upper envelope on structure: interpretability remains solid for compact encoders; absorption is low at moderate L_0 that avoids unnecessary latent proliferation; and Core is near its ceiling. SCR@20 is the most decisive among the three, pointing to cleaner isolation of spurious factors with small, targeted ablations. Sparse Probing is strong and, together with higher RAVEL, indicates that only a handful of latents capture both predictive signal and causally specific attributes with minimal collateral interference.

B.3 SAEBench Runtime Cost

The computational requirements for running SAEBench evaluations were measured on two NVIDIA RTX A800 GPUs using **16K**-width SAEs trained on the Gemma-2-2B (Team et al., 2024), Qwen-2.5-3B (Yang et al., 2024) and Gemma-2-9B. Table 4 summarizes the *per-SAE* runtime for each evaluation type. Several evaluations include a one-time setup phase (e.g., precomputing activations or training probes) that can be reused across multiple SAEs; after this setup, each evaluation has its own runtime per SAE. We therefore report amortized per-SAE minutes.

Table 4: **Approximate SAEBench runtime per SAE (minutes).** Values are per-SAE and represent amortized minutes after any one-time setup; each minute figure is an approximation and may vary with hardware and I/O.

Model	Core	Interpretability	Absorption	Sparse Probing	Ravel	SCR
Gemma-2-2B	4	8	12	2	18	10
Qwen-2.5-3B	7	9	15	8	17	16
Gemma-2-9B	11	12	17	30	40	28

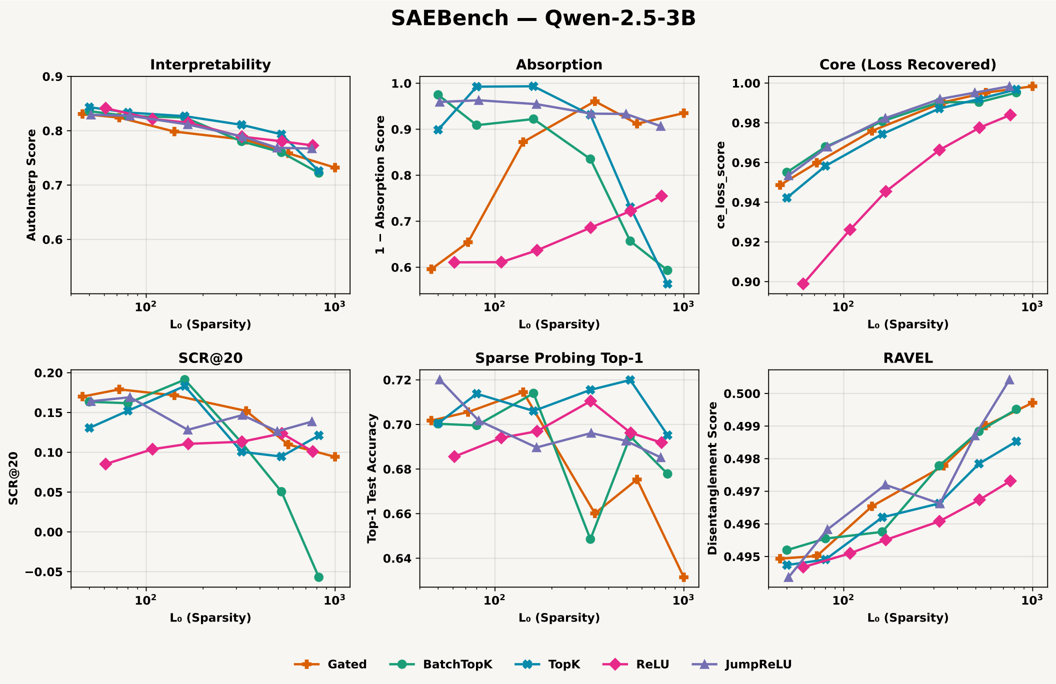

Figure 6: SAEbench results for **Qwen-2.5-3B**: strong interpretability at lower L_0, rising absorption with capacity, consistently high Core, and more fragile SCR/RAVEL at the highest capacities.

C AxBench Details and Our Costs

C.1 Steering Score Process

For each SAE feature (concept) we evaluate steering utility with an AxBench-style judge (Wu et al., 2025). Given the concept description from SAEBench and the intervention rule in Eq. equation 1, we sample 10 instructions (Alpaca-Eval style) and split them $5/5$ into a development set for factor selection and a held-out set for evaluation. We generate continuations (up to 128 tokens) for a grid of steering factors $\alpha \in \mathcal{A}$ by injecting the feature direction at the target layer. For each generated continuation, an LLM judge assigns three subscores on the discrete scale $\{0, 1, 2\}$ (floats allowed but clamped to $[0, 2]$): *Concept* (C; does the output express the concept?), *Instruction* (I; is it on-topic for the instruction?), and *Fluency* (F; is it well-formed?). We aggregate with the harmonic mean

$$\mathrm{HM}(C, I, F) = \begin{cases} 0, & \text{if } C \leq 0 \text{ or } I \leq 0 \text{ or } F \leq 0, \\ \dfrac{3}{\frac{1}{C} + \frac{1}{I} + \frac{1}{F}}, & \text{otherwise}, \end{cases} \qquad \in [0, 2].$$

On the development split we pick $\alpha^\star = \arg\max_{\alpha \in \mathcal{A}} \frac{1}{5} \sum \mathrm{HM}$, then evaluate the held-out split with $\alpha^\star$ to obtain the *per-concept Steering Score* (mean HM over the 5 held-out prompts). SAE-level utility $g(\theta)$ is the average of per-concept scores over the evaluation set for that SAE.

CONCEPT_TEMPLATE

[System]

Please act as an impartial judge and evaluate whether the specified concept is presented in the sentence fragment provided below. Focus solely on whether the concept is clearly incorporated, without regard for grammar or logical coherence.

Begin your evaluation by providing a short explanation of whether the concept is clearly incorporated. Point out references to the concept in the fragment, noting any minimal or awkward incorporations. After providing your explanation, rate the concept's relevance on a scale from 0 to 2, where 0 indicates the concept is not present at all, 1 indicates the concept is somewhat present but minimally or awkwardly incorporated, and 2 indicates the concept is more fully and effectively incorporated, with stronger and

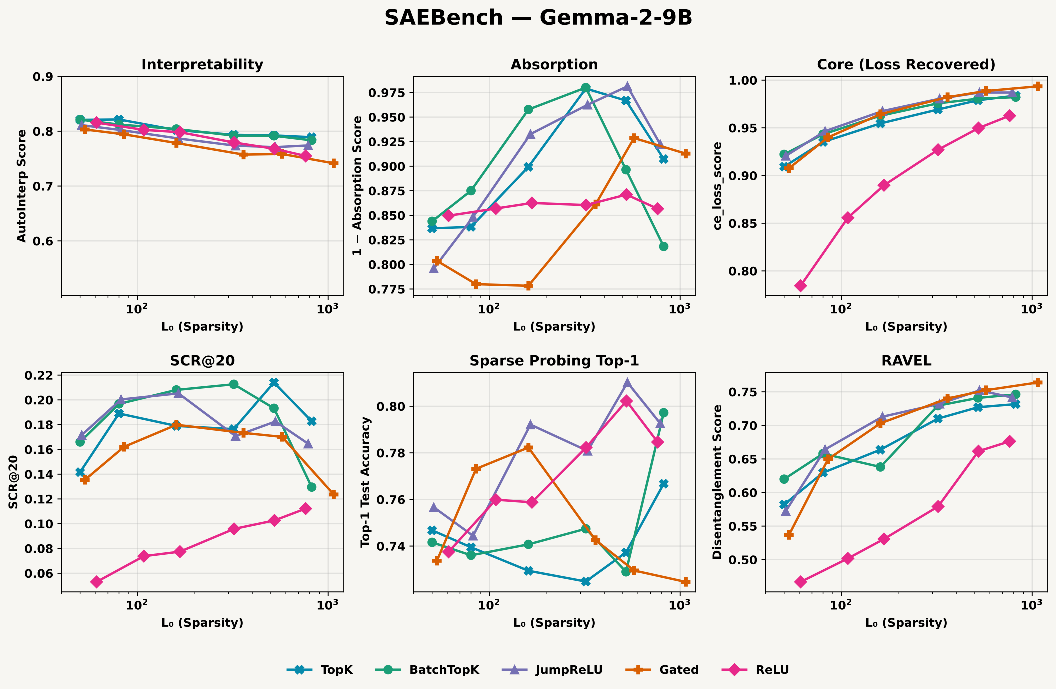

Figure 7: SAEbench results for **Gemma-2-9B**: robust interpretability with compact codes, low absorption at moderate L_0, near-ceiling Core, and the clearest gains in SCR/RAVEL among the three backbones.

more natural integration. Provide your rating using this exact format: “Rating: [[score]]”.

[Concept Start] {concept} *[Concept End]*
[Sentence Fragment Start] {sentence} *[Sentence Fragment End]*

INSTRUCT_TEMPLATE

[System]
Please act as an impartial judge and evaluate whether the sentence fragment provided below is related to the instruction. Focus solely on the degree of relatedness in terms of topic, regardless of grammar, coherence, or informativeness.
Begin your evaluation by providing a brief explanation of whether the sentence is related to the instruction, and point out references related to the instruction. After providing your explanation, rate the instruction relevance on a scale from 0 to 2, where 0 indicates the sentence is unrelated to the instruction, 1 indicates it is somewhat related but only minimally or indirectly relevant in terms of topic, and 2 indicates it is more clearly and directly related to the instruction. Provide your rating using this exact format: “Rating: [[score]]”.

[Instruction Start] {instruction} *[Instruction End]*
[Sentence Fragment Start] {sentence} *[Sentence Fragment End]*

FLUENCY_TEMPLATE

[System]
Please act as an impartial judge and evaluate the fluency of the sentence fragment provided below. Focus solely on fluency, disregarding its completeness, relevance, coherence with any broader context, or informativeness.
Begin your evaluation by briefly describing the fluency of the sentence, noting any unnatural phrasing, awkward transitions, grammatical errors, or repetitive structures that may hinder readability. After providing your explanation, rate the sentence’s fluency on a scale from 0 to 2, where 0 indicates the sentence is not fluent and highly unnatural (e.g., incomprehensible or repetitive), 1 indicates it is somewhat fluent but contains noticeable errors or awkward phrasing, and 2 indicates the sentence is fluent and almost perfect. Provide your rating using this exact format: “Rating: [[score]]”.

[Sentence Fragment Start] {sentence} *[Sentence Fragment End]*

C.2 AxBench Steering Evaluation Cost

All steering-score evaluations were run on **two NVIDIA RTX A800 GPUs**. The LLM-as-judge backend was `gpt-4o-mini`. Evaluating one SAE on Concept100 costs approximately **$5** in judge API fees; with **90** SAEs total ($\approx$ 30 per model), the per-model API cost is about **$150**. Table 5 lists approximate per-SAE runtime and peak VRAM for each model.

Table 5: **AxBench steering evaluation cost per SAE.** Runtimes are per-SAE (hours) and approximate; VRAM is peak memory (GB). Judge fees assume `gpt-4o-mini`: $\sim$ \$5 per SAE on Concept100; -Model Cost assumes $\sim$ 30 SAEs/model ($\approx$ \$150).

Model	Runtime / SAE (h)	Peak VRAM (GB)	Per-Model Cost (USD)
Gemma-2-2B	15	10	150
Qwen-2.5-3B	16	12	150
Gemma-2-9B	23	36	150

D Implementation of Δ Token Confidence

For a fixed, neutral prefix s (we use *"From my experience,"*, following the previous work(Arad et al., 2025)) we compare the next-token distribution of the base model with that of an intervened model in which a single SAE feature is amplified at layer L by a factor α. The intervention is applied via the same SAE hook point used during training (on the residual stream of block L). We then compute the change in a confidence surrogate built from the top-k probabilities.

Token confidence. (Fu et al., 2025) For a distribution p over the vocabulary, let $p_{(1)} \geq \cdots \geq p_{(k)}$ be the top-k probabilities.

$$C_k(p) \;=\; -\frac{1}{k}\sum_{i=1}^{k}\log p_{(i)}.$$

Delta token confidence. With p_{base} from a standard forward pass and p_{int} from a pass with the SAE feature intervention,

$$\Delta C_k(f;\alpha,L) \;=\; C_k\big(p_{\text{int}}\big) \;-\; C_k\big(p_{\text{base}}\big).$$

Each feature f is evaluated with two single-step forwards on the same prefix s: (i) a baseline pass; (ii) an intervened pass where we scale the code for f by α before decoding it into the residual at layer L while keeping all other codes at zero. Hooks are removed immediately after the intervened pass to prevent accumulation across evaluations. In this work, we choose $\alpha = 10$ and $k = 1$.

Feature selection from ΔC_k. For each SAE we rank its features by ΔC_k in two directions: *UP* (largest positive ΔC_k) and *DOWN* (most negative ΔC_k). We form selection sets using either (i) top-K by magnitude with $K \in \{1, 2, 3, 4, 5\}$ per direction, or (ii) upper/lower-tail quantiles (e.g., $q \in \{0.99, 0.95, 0.90, 0.80\}$ mirrored for the lower tail). These sets are then carried into AxBench (Wu et al., 2025) to measure utility lift.

E Steering Results of SAE Architectures After Feature Selection

We quantify steering with the AxBench judge after selecting features using Δ *Token Confidence* (Appendix D). Unless otherwise noted, lifts are reported as the percentage change of a given SAE's *steering score* relative to its own baseline (no selection). Results are organized at three levels: aggregate across SAEs per base model, per-SAE rankings, and distribution by architecture.

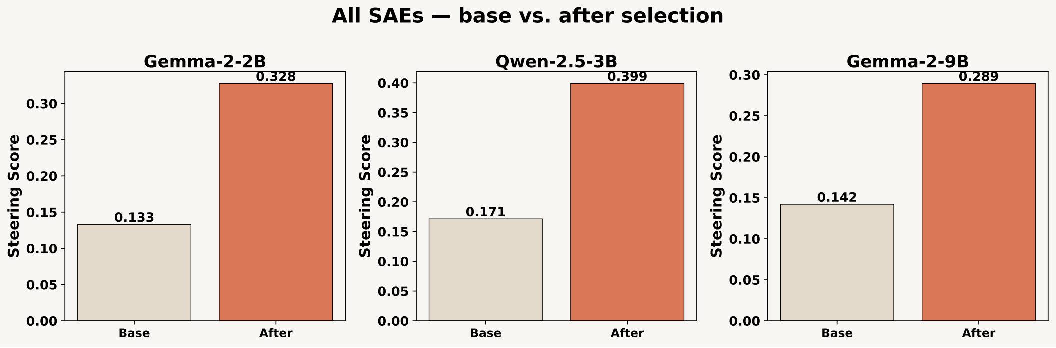

Figure 8: **Overall *steering score* before and after feature selection.** For each base model, the panel shows two bars: the average baseline *steering score* across its SAEs and the average after applying Δ Token Confidence–based selection. Bars are annotated with the corresponding values; axes share the same scale across panels.

The aggregate view in Figure 8 summarizes how selection affects the mean *steering score* across all SAEs of a base model. Using Δ *Token Confidence* for feature selection markedly improves the *steering score* across all three models in the figure. For Gemma-2-2B, the score rises from 0.133 to 0.328, which is a 146.6% relative improvement. Qwen-2.5-3B increases from 0.171 to 0.399, a 133.3% improvement, and achieves the highest post-selection score overall. Gemma-2-9B moves from 0.142 to 0.289, a 103.5% improvement.

Conclusions: (i) feature selection via Δ *Token Confidence* consistently boosts steering for all models; (ii) relative gains are largest for the smallest model (Gemma-2-2B) and smallest for the largest model (Gemma-2-9B), suggesting diminishing relative returns with scale; and (iii) in absolute terms, Qwen-2.5-3B reaches the strongest final *steering score* after selection.

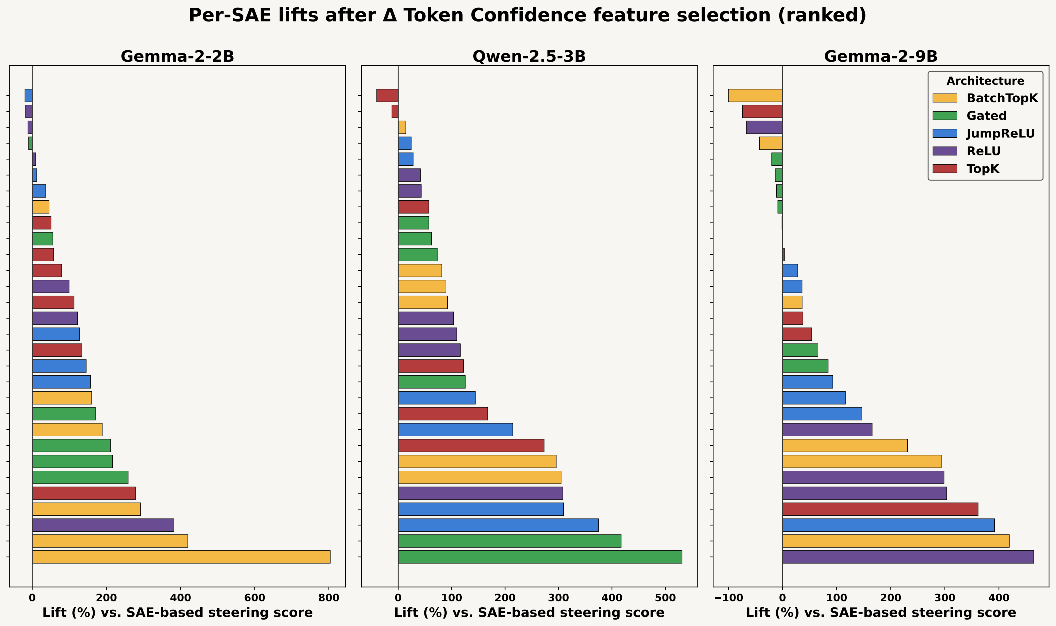

Figure 9: **Per-SAE percentage lift after Δ Token Confidence selection.** Each panel corresponds to a base model. Horizontal bars report the percent lift of the SAE-level *steering score* relative to its own baseline, sorted from largest to smallest within the panel. Bar colors indicate the SAE training architecture (legend shared across panels).

Figure 9 ranks SAEs within each model by their relative lift. Architecturally, no single SAE training approach dominates; however, the top-ranked lifts are frequently occupied by *BatchTopK* and *Gated* variants, with *ReLU*/*JumpReLU* also contributing strongly and *TopK* showing more mixed outcomes. Overall, Δ *Token Confidence* yields consistent per-SAE gains, with variance decreasing and stability increasing as model size grows, while architectural diversity remains valuable for capturing the largest lifts.

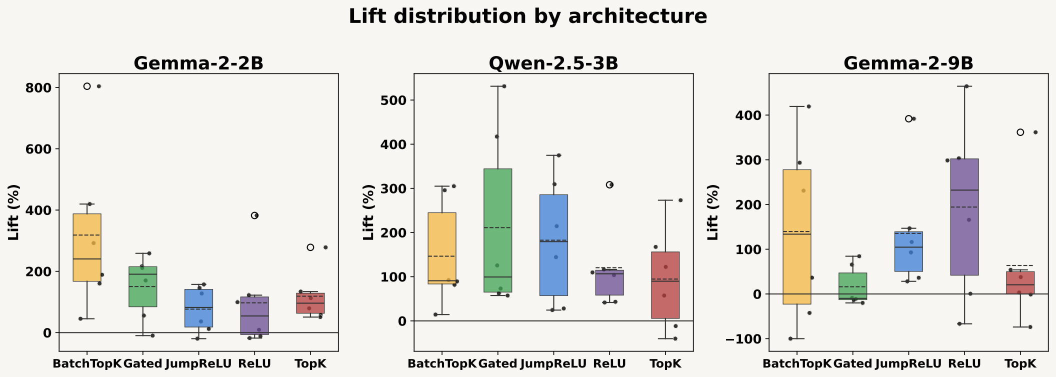

Figure 10: **Lift distributions by SAE architecture.** For each base model, box-and-whisker plots (with individual points overlaid) summarize the distribution of percentage lifts grouped by training architecture. Dashed horizontal lines denote the mean within each group, and whiskers follow the conventional interquartile rule.

Figure 10 groups lifts by architecture to visualize differences in central tendency and dispersion under the same selection and evaluation protocol. Read together with the per-SAE ranking, this distributional view helps disentangle architecture effects from model-specific variation and indicates which families tend to produce more stable or more variable outcomes after feature selection. *BatchTopK* and *Gated* generally occupy the highest central tendency with wide—but mostly positive—spread, especially on Gemma-2-2B and Qwen-2.5-3B. *BatchTopK* achieves the most stable and sizable *steering score*. Variance is largest for the smallest model (Gemma-2-2B), indicating architecture-sensitive wins at small scale.

F Dataset

Training corpus for SAEs. We train all SAEs on **The Common Pile v0.1** (Kandpal et al., 2025), an openly licensed ∼8 TB text collection built for LLM pretraining from ∼30 sources spanning research papers, code, books, encyclopedias, educational materials, and speech transcripts. The corpus was curated as a principled alternative to unlicensed web text and has been validated by training competitive 7B models on 1–2T tokens. We use it as the sole pretraining dataset for all SAE runs. More training details provided in Appendix A.2.

Concept100 for steering utility. To evaluate steering, we construct Concept100: a compact benchmark of *100 human-readable concept descriptions* per evaluation set, produced automatically by our interpretability pipeline (Appendix B). Each entry is a pair (`layer_feature_id`, `description`) that summarizes a latent's activation pattern in plain language (e.g., mathematical symbols, scientific terms, pronouns, or domain phrases). These descriptions are supplied to the AxBench judge when computing *steering score*. The examples below illustrate the style and domain coverage.

Gemma-2-9B, BatchTopK, $L_0 \approx 80$. Ten examples:

1. `20_14429`: concepts related to optical communication systems and their performance characteristics
2. `20_5795`: specific technical terms and chemical compounds often related to scientific contexts

3. `20_7908`: terms related to gravitational lensing and its effects in cosmology
4. `20_3042`: pronouns and verbs indicating relationships or contributions in various contexts
5. `20_11897`: scientific measurements and units related to energy, concentration, or biological data
6. `20_12944`: terms related to cell types and apoptosis mechanisms in scientific contexts
7. `20_8796`: references to specific authors and statistical concepts in mathematical contexts
8. `20_6430`: the phrase “action” in mathematical and theoretical contexts
9. `20_2220`: chemical elements and compounds, particularly including metals and metal-related terms
10. `20_585`: various forms of the word “energy” and related concepts in scientific contexts

Qwen-2.5-3B, Gated, $L_0 \approx 72$**.** Ten examples:

1. `17_15113`: terms related to mathematical concepts and various scientific names or terms
2. `17_11476`: the phrase “as a function of” in contexts of measurement and analysis
3. `17_162`: mathematical symbols and concepts related to coordinates, magnitudes, and parameters in equations
4. `17_2552`: dataset identifiers and technical terms common in research and academic documents
5. `17_16377`: mathematical notation and technical terms commonly found in formal documents
6. `17_3195`: demographic, clinical, and biological characteristics in study populations and related comparisons
7. `17_9186`: specific technical terms and concepts related to networking and programming
8. `17_11487`: mathematical notation and variables related to functions and equations
9. `17_14657`: mathematical notations and structures involving angle brackets and properties of functions
10. `17_1256`: terms related to errors and error correction in coding theory and quantum operations

Gemma-2-2B, JumpReLU, $L_0 \approx 81$**.** Ten examples:

1. `20_11531`: terms related to sports, programming, or specific keywords from various contexts
2. `20_10460`: terms related to fractional differential equations and numerical methods for solving them
3. `20_4882`: terms related to asymptotic theory, robustness, and statistical estimation methods
4. `20_4425`: first-person plural pronouns and expressions of intention or conjecture
5. `20_372`: technical terms related to measurement and structure in scientific contexts
6. `20_9999`: the word “from” and contexts implying deviation or distance from something
7. `20_9703`: the word “based” in various contexts of theoretical foundations and methodologies
8. `20_15509`: technical or numerical concepts in a variety of contexts

9. `20_8218`: phrases indicating conditions or assumptions that must be met in theoretical contexts
10. `20_4614`: time intervals and durations mentioned in the context of studies or observations

We currently maintain 90 SAEs (30 per base model). Beyond the CONCEPT100 sets evaluated in this paper, we have constructed the CONCEPT1000 and CONCEPT16K suites that scale the number of human-readable concepts up to 16K. We will extend training and evaluation to these larger suites in forthcoming releases to further substantiate the reliability and generality of this work.

G ANALYSIS OF STEERING STRENGTH ACROSS INTERPRETABILITY LEVELS

To directly address the concern that highly interpretable SAE features might have negligible steering effects, we conduct an additional analysis combining interpretability scores with steering scores at the feature level.

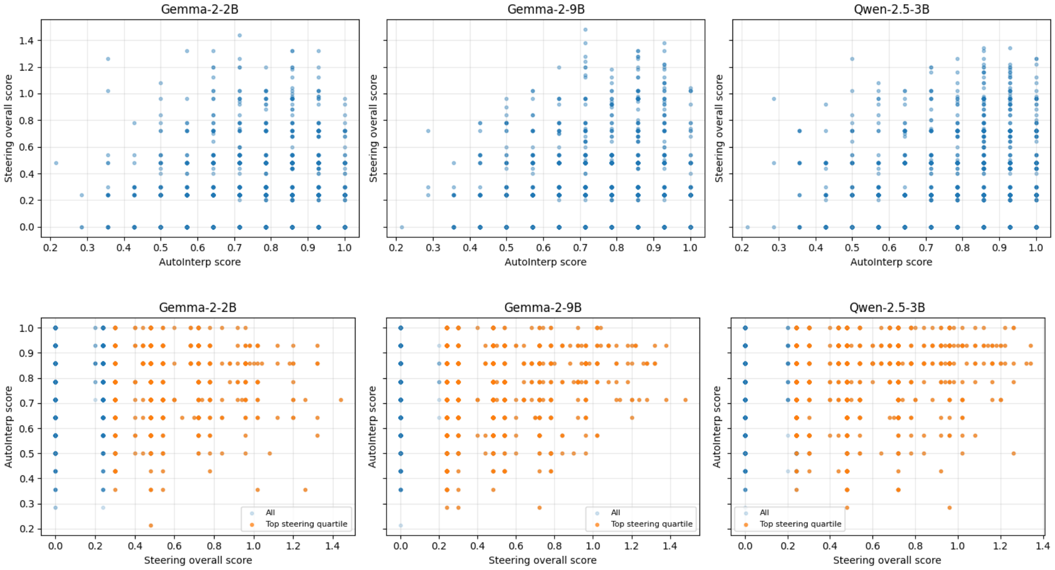

Figure 11: **Feature-level relationship between interpretability and steering strength across models.** Panels are shown separately for Gemma-2-2B, Gemma-2-9B, and Qwen-2.5-3B. The first set of panels plots interpretability score (AutoInterp) against steering overall score for every latent. The second set of panels flips the axes and highlights the top steering quartile in orange, with all latents shown in blue.

We do not find that interpretable SAE features have negligible steering effects. As the first set of panels in Figure 11 shows, high-interpretability features span the full range of steering strengths, including many strongly steering latents; high-interpretability features are not dominated by weak steering. Conversely, the second set of panels in Figure 11 shows that strongly steering features (top steering quartile) also cover almost the full range of interpretability scores, and many of them are only moderately or weakly interpretable according to interpretability score. In the SAEs under Qwen-2.5-3B, we found that features with extremely high steering scores also tend to have higher interpretability scores. This explains why, in Appendix J, only the SAE model under Qwen-2.5-3B shows improved steering performance when using high interpretability scores. However, below 1.0, the performance becomes highly dispersed again.

Taken together, these two views indicate that (i) high interpretability does not guarantee strong steering, and (ii) strong steering does not require high interpretability. In other words, feature-level interpretability scores provide limited information about how strongly a latent direction pushes the logits, reinforcing our claim that interpretability and steering strength are nearly orthogonal at the feature level.

H INSTRUCTION-FOLLOWING UNDER FEATURE-SELECTION STRATEGIES

To address whether our feature-selection metric surfaces features that make model outputs incoherent, we compare the AxBench *Instruction* subscore when steering with three strategies: using all SAE features ("SAE Base"), using features selected by the Output Score baseline, and using features selected by our Δ Token Confidence metric. For each model and SAE architecture, we aggregate the instruction subscores of the steered generations and plot them in Figure 12.

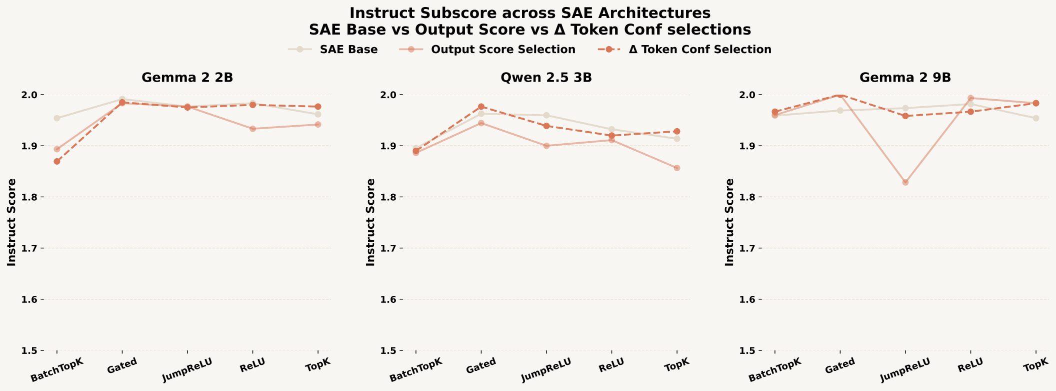

Figure 12: **Instruction subscore across SAE architectures when steering with different feature-selection strategies.** For each model, we compare the AxBench *Instruction* score under three setups: using all SAE features (SAE Base), using Output Score selection, and using Δ Token Confidence selection.

Across all models, both selection methods keep the instruction subscore close to the base level (typically in the 1.5–2.0 range on a 0–2 scale), indicating that neither Output Score nor Δ Token Confidence systematically degrades instruction following. This behavior is expected because coherence is primarily controlled by the *steering factor* α in the intervention

$$x^{\text{steer}} = x + (\alpha m_f) \cdot v_f,$$

which scales how strongly a feature f is injected into the hidden state. AxBench already tunes α per feature to balance concept shift against instruction following and fluency. The role of Δ Token Confidence is not to push the model into incoherent regimes, but to select features for which even small interventions induce a strong, directional change in the token-level logits.

I EXPLORING THE STEERING UTILITY OF SAES WITH HIGH INTERPRETABILITY

To directly test whether selecting features by interpretability also selects for high steering utility, we run a controlled comparison at the feature level. For each SAE in our three model–layer settings, we compute three steering configurations on AxBench: (i) **SAE Base**, which steers with all evaluated features; (ii) **Interpretability selection**, which steers using only the top-$p\%$ latents by AutoInterp score (we use $p = 10\%$ in Figure 13); (iii) Δ **Token Confidence selection**

Across all three models, selecting features by interpretability alone yields at best modest and often negative changes in steering strength relative to using all SAE features (about 35% for Gemma-2-2B, +10% for Qwen-2.5-3B, and 19% for Gemma-2-9B), whereas Token Confidence consistently delivers large gains of roughly 1.1–1.5× over the same SAE base. This pattern indicates that feature-level interpretability is a weak signal for steering utility, while Token Confidence reliably concentrates probability mass on features that produce substantially stronger steering effects.

J SIGNIFICANCE ANALYSIS FOR DIFFERENT SAE STEERING METHODS

To address the question of whether the improvements in Figure 4 are statistically significant, we reran the analysis at the SAE level and, for each model and architecture, computed the mean steering

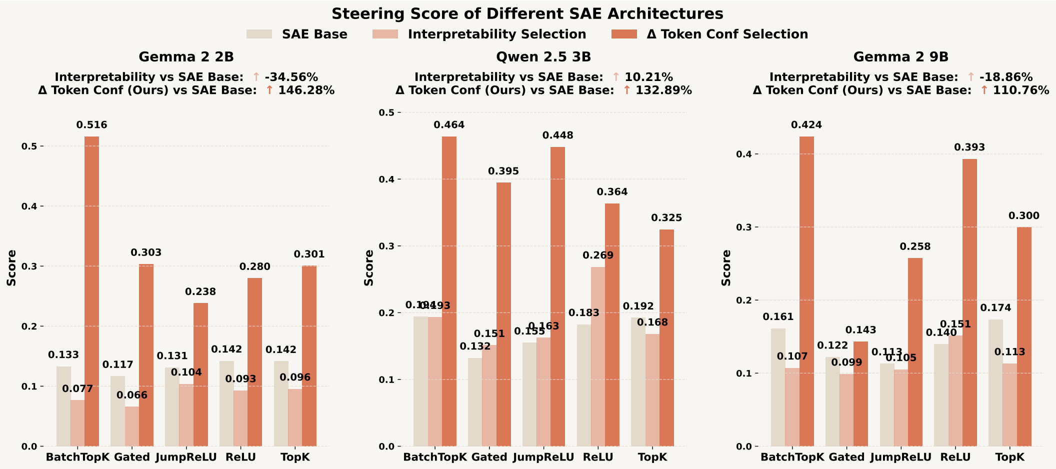

Figure 13: **Steering scores under different feature-selection strategies across SAE architectures and three models.** Bars show the AxBench overall steering score for (left) steering with all features (*SAE Base*), (middle) steering with the top-10% most interpretable features (AutoInterp), and (right) steering with features selected by Δ Token Confidence. The banners report the macro-average percentage lift over the SAE Base.

score and its 95% confidence interval across SAEs (shown as error bars in Figure 14). In addition, we performed paired bootstrap tests over SAEs for three contrasts: (i) Δ Token Confidence selection vs. SAE Base, and (ii) Δ Token Confidence selection vs. Output Score selection. Finally, we repeated the paired bootstrap test pooled over all 90 SAEs.

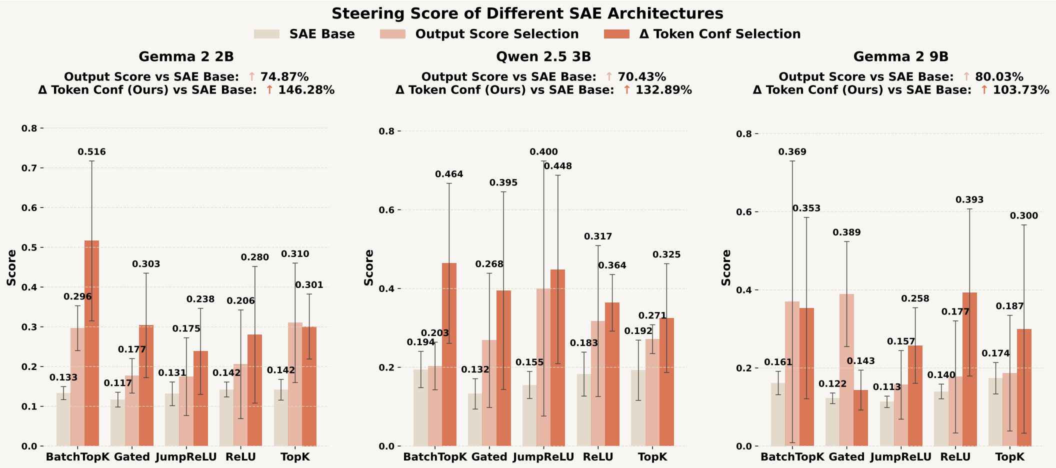

Figure 14: **Error-bar version of Figure 4.** Mean steering score and 95% confidence intervals across SAEs for the SAE Base, Output Score selection, and Δ Token Confidence selection, grouped by architecture and model. When aggregated across SAEs, there is a statistically significant difference between Δ Token Confidence and the baselines.

The error bars in Figure 14 show that Δ Token Confidence selection consistently lifts steering scores above the SAE Base across architectures and models.

In this experiment, steering scores are first computed for each of the 90 SAEs under three conditions: SAE Base (no feature selection), Output Score-based selection, and ΔTokenConf-based selection. We then form per-SAE paired differences and apply a paired bootstrap over SAEs to estimate the mean difference, its 95% confidence interval, and a two-sided p-value, yielding the global significance results summarized in Table 6.

Table 6: Pooled paired-bootstrap comparison of steering scores across all 90 SAEs. For each contrast, we report the mean per-SAE difference in steering score, a 95% bootstrap confidence interval, the two-sided p-value, and whether the effect is significant at the 5% level.

Comparison	Mean difference	95% bootstrap CI	Two-sided p	Significant at 5%?
ΔTokenConf − Base	0.1899	[0.1454, 0.2339]	≈ 0.0000	Yes
ΔTokenConf − Output Score	0.0786	[0.0261, 0.1305]	≈ 0.0032	Yes

Concretely, we find that our selection method yields a statistically significant improvement in steering score both relative to the SAE Base and relative to the Output Score selection.

K Analysis of gap between SAE Interpretability and Utility

Regarding *why there is a gap*, our view is that the root cause lies in the training paradigm of SAEs. Current SAE training is almost entirely reconstruction-centric: the objective is to accurately reconstruct internal activations while enforcing sparsity, so as to obtain a more monosemantic basis in a higher-dimensional space. This is precisely what makes SAE features interpretable, but it does not directly optimize how useful those features are for steering. In other words, the training objective is aligned with *reconstruction fidelity and monosemanticity*, not with *control over model outputs*.

To make this concrete, we perform a pairwise Kendall–τ_b analysis over all 90 SAEs, relating interpretability, steering, and two standard reconstruction-side metrics (both higher-is-better): (i) the **CE loss score** and (ii) **explained variance**. Table 7 reports τ_b, bootstrap standard error, a 95% bootstrap confidence interval, and a permutation p-value for the null $H_0 : \tau_b = 0$ for each pair of metrics.

Table 7: Pairwise Kendall–τ_b between interpretability, reconstruction metrics, and steering score over all 90 SAEs. "CE loss score" and "explained variance" are standard post-hoc reconstruction metrics (higher is better).

Pair	τ_b	SE	95% bootstrap CI	perm p ($H_0 : \tau_b = 0$)
Interpretability – CE loss score	-0.433	0.056	$[-0.539,\ -0.322]$	0.0002
Interpretability – Explained variance	-0.405	0.059	$[-0.521,\ -0.289]$	0.0002
CE loss score – Steering	-0.243	0.067	$[-0.377,\ -0.112]$	0.0006
Explained variance – Steering	-0.195	0.066	$[-0.323,\ -0.062]$	0.0062

As Table 7 shows, the reconstruction metrics exhibit only a *weak* association with steering ($|\tau_b| \approx 0.2$), but a substantially *stronger* association with interpretability ($|\tau_b| \approx 0.4$). In other words, the reconstruction-focused training objective is much more predictive of which SAEs score well under interpretability metrics than of which SAEs are actually good for steering, and is close to orthogonal to steering utility. This supports our claim that the observed interpretability–utility gap is not an artifact of our evaluation, but a structural consequence of the *current SAE training paradigm*, in which reconstruction fidelity—rather than steering behavior—is the primary optimization target.

L Ablation on the Top-k in Δ Token Confidence

In our Δ *token confidence* selection, we measure how much a feature shifts the model's confidence over the top-k tokens in the output distribution. Unless otherwise noted, all main experiments use $k = 1$, so that the score directly tracks the change in probability of the top-1 predicted token, which typically dominates next-token behavior in standard generation settings.

To verify that this choice is not artificially restrictive, we run an ablation over $k \in \{1, 3, 5, 10\}$ on Gemma-2-2B and Qwen-2.5-3B. Table 8 reports the resulting steering scores (higher is better) after applying Δ token confidence selection.

Table 8: Ablation over the top-k parameter in Δ token confidence selection.

k (top-k)	Gemma-2-2B	Qwen-2.5-3B
1	0.328	0.399
3	0.264	0.370
5	0.314	0.302
10	0.245	0.343

All tested values of k yield substantial gains over the SAE baseline (not shown), and $k = 1$ is in fact the best-performing setting on both models. We therefore adopt $k = 1$ as an empirically validated default in all main experiments.

M Examples of interpretable-but-unhelpful vs. helpful-but-hard-to-explain features

To complement our quantitative analysis, we provide qualitative examples of features where interpretability and steering utility diverge. Type A features are *highly interpretable but not steerable* (high interpretability score, near-zero steering score), while Type B features are *moderately interpretable but strongly steerable*. All generations in Table 9 are produced with temperature 0 using the fixed prefix ``` ``I heard that'' ``` and intervening on a single feature.

Table 9: Qualitative examples of features where interpretability and steering diverge. Type A: interpretable but not steerable (interpretability ≈ 1.0, steering ≈ 0). Type B: moderately interpretable but clearly steerable. All generations use temperature 0 with the prefix “I heard that”.

Field	**Gemma 2 2B (A)**	**Qwen 2.5 3B (A)**	**Gemma 2 2B (B)**	**Qwen 2.5 3B (B)**
Type	A	A	B	B
Latent id	6168	12527	589	2392
Architecture	BatchTopK, L_0=320	BatchTopK, L_0=160	ReLU, L_0=156	JumpReLU, L_0=166
Interpretability	1.00	1.00	0.57	0.50
Steering score	0.00	0.00	1.32	1.02
Explanation	the symbol ‘pi’ in mathematical expressions	terms related to differentiation in mathematical contexts	words related to actions or processes in various contexts	numerical values related to comparisons
After steering	I heard that there are a lot of people who live with their own thoughts in their heads.	I heard that it is Math and Science, did you hear that it is Math and Science?	I heard that The Body Shop **is planning to open** in Central World, I **used to go** to Bangna branch.	I heard that **a little bit larger than twice the actual, 2000 feet (2,500 meters)** worth of this, which is a mile of it, **three times** it’s actual.