RATIONAL SPARSE AUTOENCODER

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse autoencoders (SAEs) are standard tools for mechanistic interpretability, but current SAE families are constrained by fixed encoder nonlinearities such as ReLU, JumpReLU, and TopK. This hard-codes a particular sparsity mechanism into the model and can distort the reconstruction-versus-sparsity trade-off. We introduce the *Rational Sparse Autoencoder* (RSAE), which replaces the fixed encoder activation with a trainable rational function. The design is motivated by a reconstruction advantage that we prove for RSAE at matched expected sparsity, with guaranteed non-inferiority to ReLU and, under stated assumptions, provable improvements over JumpReLU and TopK. The model is then implemented through a two-stage pipeline: an initialization procedure that copies the pre-trained baseline SAE weights and calibrates the scale parameters along with the rational coefficients; followed by a fine-tuning step under a sparsity-regularized reconstruction objective that holds ℓ_0 at the teacher's value. Consistent with this theory, on residual-stream activations of three open-weight language models and across all three baseline activation families, the RSAE *improves* reconstruction fidelity after the fine-tuning step at the same level of sparsity. These gains are consistent across host language models, across baseline activation families, and across the full range of baseline sparsity we tested, while the upgrade itself adds only 20 scalar parameters per autoencoder and one training run takes between about 4 and 25 minutes on a single consumer GPU.

1 INTRODUCTION

Sparse autoencoders (SAEs) have become a central tool for mechanistic interpretability of large language models, decomposing the internal activations of a transformer into a sparse linear combination of an overcomplete dictionary of monosemantic feature directions (Bricken et al., 2023; Huben et al., 2024; Gao et al., 2025; Rajamanoharan et al., 2024a;b). Despite rapid progress on training procedures and evaluation suites, the widely used released baselines considered here pair the same affine encoder with one of three non-smooth activation primitives: ReLU regularized by an ℓ_1 penalty (Bricken et al., 2023; Bloom, 2024), TopK (Gao et al., 2025), or $\mathrm{JumpReLU}$ (Rajamanoharan et al., 2024b). Each of these primitives carries well-documented pathologies. The ℓ_1-regularized ReLU SAE suffers from magnitude shrinkage of active features and a persistent population of dead latents (Taggart, 2024; Rajamanoharan et al., 2024a; Gao et al., 2025). TopK replaces the soft penalty with a hard cardinality constraint that breaks gradient flow through inactive features and relies on auxiliary revival losses to mitigate dead features. $\mathrm{JumpReLU}$ inserts a learnable per-feature threshold but requires a continuous-relaxation surrogate for back-propagation through its indicator gate.

In this work we propose to replace these pre-selected activation functions with a trainable one, which we call the *Rational Sparse Autoencoder* (RSAE). In particular, the encoder activation in the RSAE is a learnable rational function applied element-wise to the affine pre-activation, with learnable input/output scales $(C_{\mathrm{in}}, C_{\mathrm{out}})$ that map the per-feature pre-activation distribution into a bounded interval. This architecture is supported by a new analysis: we provide reconstruction guarantees against all three baseline activations, showing that the RSAE exactly represents every ReLU SAE, preserving its code and reconstruction and yielding same-budget non-inferiority (Appendix A.1). Under Stability and Nonlinear-calibration persistence, shared rational calibration gives an $\Omega(k)$ reconstruction improvement over JumpReLU with per-feature thresholds while preserving its support (Appendix A.3). For fixed raw TopK, the same calibration argument applies after a support replace-

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

ment that costs $O(\sqrt{k})$, so the gain dominates for sufficiently large sparse budgets (Appendix A.2). Each comparison preserves the decoder and latent width and matches the expected $\ell_0 = k$.

For practitioners, we propose a two-step RSAE training algorithm. During the initialization procedure, we copy the pre-trained baseline SAE weights verbatim and calibrate the scale parameters and the coefficients to the baseline's pre-activation distribution. During the fine-tuning procedure, we unfreeze all parameters and minimize a sparsity-regularized reconstruction objective that holds the expected ℓ_0 at the teacher's value. A single low-degree rational of type $(9, 8)$ reproduces a ReLU teacher exactly and a JumpReLU or TopK teacher to within 1–3% of its reconstruction error before fine-tuning. At the SAE level, we evaluate the RSAE on residual-stream activations of three open-weight language models spanning a range of model sizes and against all three baseline activation families, supporting our central claim: at matched ℓ_0, the RSAE improves on its teacher in reconstruction error, cross-entropy degradation, loss recovered and, in almost all settings, the fraction of alive features, across host language models and baseline activation families. A control that freezes the rational shows that most of the gain on ReLU teachers comes from the trainable activation itself. These reconstruction fidelity gains are observed across the full range of baseline sparsity we tested, and their relative size grows with ℓ_0. A sparse-probing sanity check (Appendix B.3) shows that they do not come at the cost of probe accuracy. Each RSAE adds only 20 scalar parameters to its teacher and is trained in at most about 25 minutes on a single consumer GPU.

Contributions. We introduce *RSAE*, a new sparse autoencoder built on a trainable activation and supported by theoretical guarantees on reconstruction fidelity. We prove ReLU non-inferiority and, under Stability and Nonlinear-calibration persistence, reconstruction advantages over JumpReLU with per-feature thresholds and over fixed raw TopK for sufficiently large sparse budgets, at matched expected ℓ_0. Practically, to implement this upgrading strategy, we propose a two-step RSAE training algorithm: an initialization procedure that copies the pre-trained baseline SAE weights, followed by a fine-tuning procedure that unfreezes all parameters under a sparsity-regularized reconstruction objective at the teacher's ℓ_0. We show empirically that the RSAE achieves better fidelity at matched sparsity with negligible parameter and runtime overhead: it improves on its teacher in reconstruction error, cross-entropy degradation and loss recovered across host language models, baseline activation families, and baseline sparsity levels.

2 Preliminaries and Related Work

Sparse Autoencoders (SAEs) decompose a language model's internal activations $\boldsymbol{x} \in \mathbb{R}^{d_{\text{in}}}$ into a sparse linear combination of an overcomplete dictionary of $d_{\text{sae}} \gg d_{\text{in}}$ feature directions $\boldsymbol{z} \in \mathbb{R}^{d_{\text{sae}}}$. They follow a skeleton with a pair of encoder and decoder functions (f, g) defined by:

$$\text{Encoder: } \boldsymbol{z} = f(\boldsymbol{x}) := \phi(\boldsymbol{W}_{\text{enc}}\,(\boldsymbol{x} - \boldsymbol{b}_{\text{dec}}) + \boldsymbol{b}_{\text{enc}}), \quad \text{Decoder: } \hat{\boldsymbol{x}} = g(\boldsymbol{z}) := \boldsymbol{W}_{\text{dec}}\,\boldsymbol{z} + \boldsymbol{b}_{\text{dec}}. \quad (1)$$

We write $\boldsymbol{h} \coloneqq \boldsymbol{W}_{\text{enc}}(\boldsymbol{x} - \boldsymbol{b}_{\text{dec}}) + \boldsymbol{b}_{\text{enc}}$ for the pre-activation, so that $\boldsymbol{z} = \phi(\boldsymbol{h})$. Here, columns of $\boldsymbol{W}_{\text{dec}}$ represent decoder dictionary directions used to reconstruct $\boldsymbol{x}$ from the sparse code $\boldsymbol{z}$ with unit ℓ_2-norm. The weights in encoder/decoder functions are optimized using a loss function of the form:

$$\mathcal{L}(\boldsymbol{W}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\Big[\, \big\|\boldsymbol{x} - \hat{\boldsymbol{x}}(\boldsymbol{x};\, \boldsymbol{W})\big\|_2^2 \;+\; \lambda\, S\big(\boldsymbol{z}(\boldsymbol{x};\, \boldsymbol{W})\big)\,\Big], \quad \boldsymbol{W} := \{\boldsymbol{W}_{\text{enc}}, \boldsymbol{W}_{\text{dec}}, \boldsymbol{b}_{\text{enc}}, \boldsymbol{b}_{\text{dec}}\}, \quad (2)$$

where S is a function that penalizes non-sparse decompositions with a tunable sparsity coefficient λ.

There are two objectives in the SAE encoder: sparsity, meaning that only a few elements of the dictionary are necessary, and faithfulness, meaning that the reconstructed $\hat{\boldsymbol{x}}$ is close to the original $\boldsymbol{x}$. To achieve a good balance between these two objectives, three major SAE activations were proposed, which differ in the encoder activation ϕ and the sparsity mechanism S imposed on $\boldsymbol{z}$. The *ReLU SAE* (Bricken et al., 2023; Bloom, 2024) sets $\phi = \text{ReLU}$ and imposes sparsity through an explicit ℓ_1 penalty $S(\boldsymbol{z}) := ||\boldsymbol{z}||_1$. In the original *ReLU SAE*, the soft ℓ_1 penalty leads to a magnitude shrinkage of active features and causes loss of reconstruction fidelity (Taggart, 2024; Rajamanoharan et al., 2024a; Gao et al., 2025). *TopK SAE* (Gao et al., 2025) then proposes to replace the soft penalty with a hard top-k selection $\boldsymbol{z} = \text{TopK}_k(\boldsymbol{h})$ that yields exact $\ell_0 = k$, and the *JumpReLU SAE* (Rajamanoharan et al., 2024b) penalizes ℓ_0 directly, using straight-through estimators to train through the discontinuity, and inserts a learnable per-feature threshold $\theta_j > 0$ in the activation function ϕ by setting $\phi(\boldsymbol{h}) = \boldsymbol{h} \odot H(\boldsymbol{h} - \boldsymbol{\theta})$, where H is the Heaviside function satisfying $H(z) = 0$ if $z \leq 0$ and $H(z) = 1$ elsewhere. Orthogonal to the development in encoder activations, *Matryoshka*

SAEs (Bussmann et al., 2025) reorganize the decoder into nested-prefix dictionaries, and *data-free SAEs* (Laptev et al., 2025) fit dictionaries directly from model weights without streaming activations.

While related variants such as the *Gated SAE* (Rajamanoharan et al., 2024a), ProLU (Taggart, 2024), *BatchTopK SAE* (Bussmann et al., 2024), and end-to-end SAE training (Braun et al., 2024) modify thresholds, gates, batch-level sparsity, or the training objective, our theoretical results and empirical comparisons focus on the widely used released baselines considered here—ReLU, JumpReLU, and TopK SAEs—whose encoder nonlinearities are fixed functional forms with sparsity controlled by a penalty coefficient, learned threshold, or cardinality budget rather than by a trainable rational activation. Through a trainable activation architecture supported by theoretical guarantees, our RSAE provides a drop-in modification for pre-trained SAEs (teacher models): at matched expected ℓ_0, it is never worse than ReLU and, under stated assumptions, strictly improves reconstruction fidelity over JumpReLU and fixed raw TopK.

Evaluation benchmarks and pretrained baselines. *SAEBench* (Karvonen et al., 2025) provides matched pretrained ReLU, JumpReLU, and TopK SAEs across model sizes and a unified evaluation suite (covering reconstruction, sparsity, downstream performance, and interpretability metrics); we use the pre-trained ReLU, JumpReLU, and TopK SAEs for our Pythia-160m and Gemma-2-2B baselines, and its core evaluation protocol and sparse-probing harness in §5. For GPT-2 small we additionally use Bloom's *gpt2-small-res-jb* release (Bloom, 2024) as the ReLU baseline and OpenAI's v5 release (Gao et al., 2025) as the TopK baseline.

Rational Neural Networks were built on the key theoretical advantages of rational functions in approximating non-smooth functions (Newman, 1979; Telgarsky, 2017; Beckermann & Townsend, 2017; Chen et al., 2018). In Boullé et al. (2020), a rational function is employed as a learnable replacement for ReLU or $\tanh$ in feed-forward networks for image classification tasks. Superior performance was also demonstrated in operator learning and PDE surrogates, where the spectral density of rational approximation accelerates convergence on smooth target operators (Trimmel et al., 2022). In rational neural networks, the standard activation function in a feed-forward layer is replaced with a trainable rational function $\frac{P(t)}{Q(t)}$. A naive learnable denominator $Q(t) = 1 + \sum_{j=1}^{q} b_j\, t^j$ can develop divergent poles during training when Q approaches zero. To prevent this, Molina et al. (2020) introduced the Padé Activation Unit (PAU) with the *safe* denominator $Q(t) = 1 + |\sum_{j=1}^{q} b_j\, t^j| \geq 1$, which guarantees pole-free rational activations; the variant $1 + \sum_{j=1}^{q} |b_j|\, |t|^j$ used in (3) is likewise bounded below by one (Molina et al., 2020; Delfosse et al., 2024).

While prior works have demonstrated the theoretical advantages of rational activation functions against continuous activation functions such as ReLU, GeLU, or $\tanh$ (Boullé et al., 2020; Molina et al., 2020; Delfosse et al., 2024; Trimmel et al., 2022; Tang & Townsend, 2026), little discussion has focused on discontinuous SAE gates such as $\mathrm{JumpReLU}$ and TopK, or on approximation gains under fixed sparsity. Herein, we establish ReLU non-inferiority by exact representation and analyze nonlinear-calibration gains against JumpReLU and fixed raw TopK. These SAE-level comparisons retain the full decoder Gram interaction and match the expected ℓ_0 exactly.

3 Rational Sparse Autoencoder

The *Rational Sparse Autoencoder* (RSAE) replaces the encoder activation in (1) with a trainable rational function followed by a non-negativity clamp. For pre-activations $\boldsymbol{h} = \boldsymbol{W}_{\text{enc}}(\boldsymbol{x} - \boldsymbol{b}_{\text{dec}}) + \boldsymbol{b}_{\text{enc}}$,[1] the sparse code is

$$\boldsymbol{z} = \phi(\boldsymbol{h}) = C_{\text{out}} \left[r_{(\boldsymbol{a},\boldsymbol{b})}\left(\frac{\boldsymbol{h}}{C_{\text{in}}} \right) \right]_+, \qquad r_{(\boldsymbol{a},\boldsymbol{b})}(t) = \frac{P(t)}{Q(t)} = \frac{\sum_{i=0}^{p} a_i t^i}{1 + \sum_{j=1}^{q} |b_j|\, |t|^j}, \tag{3}$$

where $[u]_+ = \max(0, u)$ is applied element-wise and p and q are the numerator and denominator degrees. The denominator uses the safe (pole-free) form: $b_0 = 1$ and the absolute values guarantee $Q(t) \geq 1$ for every t and every choice of $\boldsymbol{b}$, so $r_{(\boldsymbol{a},\boldsymbol{b})}$ has no poles on the real line and can be initialized and fine-tuned without any constraint on the coefficients. The rational coefficients $(\boldsymbol{a}, \boldsymbol{b})$ and the positive scalar scales $C_{\text{in}}, C_{\text{out}}$ (parameterized through their logarithms) are trainable and

[1] We follow the input convention of the teacher: the SAEBench Standard (ReLU) and TopK SAEs subtract $\boldsymbol{b}_{\text{dec}}$ before the encoder, whereas the SAEBench JumpReLU SAEs use $\boldsymbol{h} = \boldsymbol{W}_{\text{enc}}\boldsymbol{x} + \boldsymbol{b}_{\text{enc}}$; the RSAE inherits whichever convention its teacher uses.

shared across all features; the support of the sparse code, $\{\boldsymbol{h} : \boldsymbol{z} > 0\}$, is therefore determined solely by the sign of $P(\boldsymbol{h}/C_{\text{in}})$ and involves no threshold or gating parameter. The sparse code is decoded by $\hat{\boldsymbol{x}} = \boldsymbol{W}_{\text{dec}}\boldsymbol{z} + \boldsymbol{b}_{\text{dec}}$.

We compare reconstruction at the same expected support size $k = \mathbb{E}\|z^{\text{base}}\|_0$, where z^{base} is the baseline code. Write $\mathcal{R}(z) = \mathbb{E}\|\boldsymbol{x} - \boldsymbol{W}_{\text{dec}}z - \boldsymbol{b}_{\text{dec}}\|_2^2$ for the reconstruction risk. For ReLU, containment directly yields same-budget non-inferiority.

Theorem 1 (ReLU non-inferiority)**.** *For $p \geq 1$ and $q \geq 0$, the infimum of RSAE reconstruction risk is no greater than that of ReLU at any feasible expected ℓ_0 budget, under the same encoder and decoder parameter constraints. Non-inferiority also holds for the regularized objective* (6)*, for every $\lambda \geq 0$ and every regularizer that depends only on the code, including ℓ_1.*

For fixed raw TopK and JumpReLU with positive per-feature thresholds, reconstruction improvements follow under the next two assumptions.

Assumption 1 (Stability)**.** *The mean squared active-feature magnitude is uniformly bounded. For multiplicative perturbations defined by a bounded shared function of the pre-activation, expected decoded energy is at most a fixed constant times expected coefficient energy. For TopK, a common threshold matching the expected count k gives count variance $O(k)$, and successive reconstruction gains in descending score order have uniformly bounded moments of some order greater than two about a common deterministic center.*

Assumption 2 (Nonlinear-calibration persistence)**.** *After optimal shared rescaling, the product of an active coefficient and the residual's projection onto its decoder direction has a second moment bounded away from zero and a bounded normalized fourth moment. Its conditional mean given the scalar pre-activation captures a nonvanishing fraction of this second moment.*

The quantitative conditions are stated in Assumptions A.2 and A.3.

Theorem 2 (Same-budget reconstruction advantage)**.** *Under Assumptions 1 and 2, in the settings of Theorems A.4 and A.7, there are constants $c \geq 0$ and $g > 0$, depending only on the assumption bounds, such that for every $\varepsilon > 0$ there exists an RSAE with a finite-degree shared rational activation without real poles, the same decoder and latent width, and $\mathbb{E}\|z^{\text{Rat}}\|_0 = k$, satisfying*

$$\mathcal{R}(z^{\text{base}}) - \mathcal{R}(z^{\text{Rat}}) \geq \begin{cases} gk - c\sqrt{k} - \varepsilon, & \text{base} = \text{TopK}, \\ gk - \varepsilon, & \text{base} = \text{JumpReLU}. \end{cases} \tag{4}$$

The encoder is unchanged for TopK; an affine reparameterization preserves the per-feature support for JumpReLU.

Consequently, at the same expected sparsity, RSAE attains lower reconstruction risk than JumpReLU and, for sufficiently large k, fixed raw TopK. The constants g and c depend on the instance through the assumption bounds, so the bound identifies the regime that favors the RSAE (budgets k that are large relative to the support-replacement cost) rather than a fixed absolute rate of improvement in k.

Proofs are given in Appendices A.1, A.2, and A.3.

4 Practical Algorithm

The RSAE is constructed in two steps (Algorithm 1, Appendix B.1): (i) an *initialization procedure* that inherits the teacher's encoder and decoder, sets the rational coefficients $(\boldsymbol{a}, \boldsymbol{b})$ in closed form so that the rational activation reproduces the teacher's activation, and adapts the learnable scales $(C_{\text{in}}, C_{\text{out}})$ together with $(\boldsymbol{a}, \boldsymbol{b})$ to the teacher's pre-activation distribution; and (ii) a *fine-tuning procedure* that jointly optimizes all parameters, including the encoder and decoder weights, under a sparsity-regularized reconstruction objective.

Step 1: RSAE Initialization Procedure. We first initialize the encoder and decoder with the pre-trained teacher weights $\{\widetilde{\boldsymbol{W}}_{\text{enc}}, \widetilde{\boldsymbol{b}}_{\text{enc}}, \widetilde{\boldsymbol{W}}_{\text{dec}}, \widetilde{\boldsymbol{b}}_{\text{dec}}\}$, and set the scales to $C_{\text{in}} = C_{\text{out}} = \text{std}(\boldsymbol{h})$ on a calibration batch. Two closed-form initializations of the rational coefficients are used, depending on the teacher's activation.

Identity-init (ReLU teachers). We set the rational to the identity map,

$$\boldsymbol{a}^* = (0,\, 1,\, 0,\, \ldots,\, 0), \qquad \boldsymbol{b}^* = (1,\, 0,\, \ldots,\, 0),$$

so that, with the non-negativity clamp of (3) and $C_{\text{in}} = C_{\text{out}}$, $r_{(\boldsymbol{a}^*,\boldsymbol{b}^*)}(\cdot)$ is exactly the ReLU activation and the RSAE coincides with the teacher at initialization.

Knee-init (JumpReLU and TopK teachers). A gated teacher computes $\phi_j^{\text{teacher}}(\boldsymbol{h}) = h_j\,\mathbf{1}[h_j > \tilde{\theta}_j]$ with a per-feature threshold $\tilde{\theta}_j$ inherited from the teacher[2]. A shared rational cannot represent per-feature jump heights, so we first absorb $\tilde{\theta}_j$ into the inherited weights by the function-preserving rescaling $\widetilde{\boldsymbol{W}}_{\text{enc}}[:,j] \leftarrow \widetilde{\boldsymbol{W}}_{\text{enc}}[:,j]/\tilde{\theta}_j$, $\widetilde{b}_{\text{enc},j} \leftarrow \widetilde{b}_{\text{enc},j}/\tilde{\theta}_j - 1$, $\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:] \leftarrow \tilde{\theta}_j \widetilde{\boldsymbol{W}}_{\text{dec}}[j,:]$ (Appendix B.2), after which every feature computes the same unit jump $(v+1)\,\mathbf{1}[v>0]$ of its rescaled pre-activation v. We then initialize the rational as the lowest-degree knee that reproduces this jump,

$$r_{(\boldsymbol{a}^*,\boldsymbol{b}^*)}(t) = \frac{a_1 t + a_2 t^2 + a_3 t^3}{1 + K^2 t^2},$$

with (a_1, a_2, a_3) fixed by three conditions: (i) the tail matches the jump, $C_{\text{out}}\, r(\frac{v}{C_{\text{in}}}) = v + 1 + \mathcal{O}(\frac{1}{v})$, giving $a_3 = K^2 \frac{C_{\text{in}}}{C_{\text{out}}}$ and $a_2 = \frac{K^2}{C_{\text{out}}}$; (ii) the knee has width $\approx \frac{1}{K}$, giving $a_1 \geq \frac{K}{C_{\text{out}}}$; and (iii) $r(t) < 0$ for all $t < 0$, so that the clamp is exactly zero off the teacher's support, which holds iff $a_2^2 < 4a_1a_3$. We take $a_1 = \max\big(\frac{K}{C_{\text{out}}},\, 1.05\,\frac{a_2^2}{4a_3}\big)$ and $K = 30$ (JumpReLU) or 15 (TopK).

With the initialized weights and coefficients, we adapt the rational activation to the teacher's pre-activation distribution by minimizing

$$(\widetilde{\boldsymbol{a}}, \widetilde{\boldsymbol{b}}, \widetilde{C}_{\text{in}}, \widetilde{C}_{\text{out}}) \;=\; \arg\min_{\boldsymbol{a},\boldsymbol{b},C_{\text{in}},C_{\text{out}}} \big\|\phi(\boldsymbol{h};\boldsymbol{a},\boldsymbol{b},C_{\text{in}},C_{\text{out}}) - \phi^{\text{teacher}}(\boldsymbol{h})\big\|_2^2 \;+\; \mu\, S\big(\boldsymbol{z}(\boldsymbol{h};\boldsymbol{a},\boldsymbol{b},C_{\text{in}},C_{\text{out}})\big), \tag{5}$$

where $\phi(\boldsymbol{h};\boldsymbol{a},\boldsymbol{b},C_{\text{in}},C_{\text{out}})$ is computed via (3) with the encoder and decoder frozen, $\boldsymbol{h}$ is the teacher's pre-activation, and $S(\cdot)$ is a differentiable sparsity regularizer on the sparse code with coefficient $\mu \geq 0$, which keeps the support of the rational aligned with the teacher's while the first term fits the magnitudes. For gated teachers we run (5) from both initializations and keep the better fit. Combined with the inherited encoder and decoder weights, the learned coefficients and scales allow the RSAE to reproduce the teacher's output, up to the Step 1 approximation error.

Step 2: RSAE Fine-Tuning Procedure. We initialize the RSAE encoder and decoder with the teacher weights $\{\widetilde{\boldsymbol{W}}_{\text{enc}}, \widetilde{\boldsymbol{b}}_{\text{enc}}, \widetilde{\boldsymbol{W}}_{\text{dec}}, \widetilde{\boldsymbol{b}}_{\text{dec}}\}$, and the rational activation with the learned coefficients $(\widetilde{\boldsymbol{a}}, \widetilde{\boldsymbol{b}})$ and scales $(\widetilde{C}_{\text{in}}, \widetilde{C}_{\text{out}})$. We then unfreeze all parameters $\Theta := \{\boldsymbol{W}_{\text{enc}}, \boldsymbol{b}_{\text{enc}}, \boldsymbol{W}_{\text{dec}}, \boldsymbol{b}_{\text{dec}}, \boldsymbol{a}, \boldsymbol{b}, C_{\text{in}}, C_{\text{out}}\}$ and minimize the sparsity-regularized reconstruction objective

$$\min_{\Theta}\;\; \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\Big[\,\big\|\boldsymbol{x} - \hat{\boldsymbol{x}}(\boldsymbol{x};\Theta)\big\|_2^2\Big] \;+\; \lambda\, S\big(\boldsymbol{z}(\boldsymbol{x};\Theta)\big), \tag{6}$$

with sparsity coefficient $\lambda > 0$. Because the RSAE has no threshold or gating parameters, its support is determined solely by the root of the rational, so S controls sparsity only through the rational coefficients, the scales and the encoder. The choice of S (a target-ℓ_0 regularizer in our experiments; an ℓ_1 penalty recovers the classical objective), the calibration of its target, and all optimization details are deferred to Appendix B.2.

5 Empirical Results

Models, SAEs, and Evaluation Metrics. We evaluate the RSAE on residual-stream activations from three open-weight language models of various sizes: **GPT-2 small**, **Pythia-160m-deduped**, and **Gemma-2-2B**. Teacher SAEs are taken from publicly released checkpoints: GPT-2 small from Bloom's `gpt2-small-res-jb` (Bloom, 2024) (ReLU) and the OpenAI-v5 (Gao et al., 2025) `gpt2-small-resid-post-v5-32k` (TopK); Pythia-160m and Gemma-2-2B from SAEBench (Karvonen et al., 2025) for ReLU, JumpReLU, and TopK. Following the SAEBench core evaluation protocol (Karvonen et al., 2025), we evaluate our RSAE against the SAE baselines in terms of five metrics: (1) the reconstruction error, the mean over tokens of $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2$; (2) sparsity, the

[2]JumpReLU teachers learn $\tilde{\theta}_j$ during training. A TopK teacher has no threshold: it keeps the k largest pre-activations per token. Its SAEBench release stores a scalar inference threshold, the mean over tokens of the k-th largest pre-activation, with which the SAE can be run as a JumpReLU-style gate at the same mean ℓ_0; we use this value as $\tilde{\theta}_j \equiv \tilde{\theta}$ for all features (estimating it on a calibration batch if a checkpoint lacks it). Teachers themselves are always evaluated with exact top-k selection.

Table 1: **Main results:** teacher SAEs and the RSAEs fine-tuned from them (RSAE (*teacher*-init)) on residual-stream activations of GPT-2 small (layer 6), Pythia-160m (layer 8) and Gemma-2-2B (layer 12). GPT-2 small teachers are the public releases: Bloom's `gpt2-small-res-jb` ReLU SAE ($d_{\text{sae}} = 24{,}576$, ℓ_1-trained, $\ell_0 \approx 51$, hook `resid_pre`) and OpenAI's `gpt2-small-resid-post-v5-32k` TopK SAE ($d_{\text{sae}} = 32{,}768$, $k = 32$, hook `resid_post`); no JumpReLU SAE has been released for GPT-2 small. Pythia-160m and Gemma-2-2B teachers are the SAEBench releases at $d_{\text{sae}} = 2^{12}$, at the $\ell_0 \approx 160$ level for Pythia-160m and $\ell_0 \approx 80$ for Gemma-2-2B (all Pythia-160m levels and widths are reported in Table 5). Metrics: MSE, the mean per-token squared reconstruction error; ℓ_0, the mean number of non-zero latents per token; alive, the fraction of alive latents **in percent**; ΔCE, the cross-entropy degradation; LR, the loss recovered **in percent**. For every RSAE we run three independent fine-tuning trials with different random seeds (data order and stochastic training) and report the mean $\pm$ standard deviation across the three trials; teacher rows are deterministic evaluations of the released checkpoints. **Bold** marks the better value between teacher and RSAE within each pair; ℓ_0 is not bolded.

SAE	ℓ_0	MSE ($\times 100$)$\downarrow$	alive (%)$\uparrow$	ΔCE$\downarrow$	LR (%)$\uparrow$
GPT-2 small (layer 6)					
ReLU teacher	53	5.97	89.5	0.180	97.2
RSAE (ReLU-init)	51$\pm$0.3	**5.30**$\pm$0.001	**91.6**$\pm$0.1	**0.123**$\pm$0.003	**98.1**$\pm$0.02
TopK teacher	32	6.71	71.3	0.136	97.9
RSAE (TopK-init)	31$\pm$0.1	**5.96**$\pm$0.007	**71.9**$\pm$0.1	**0.092**$\pm$0.000	**98.6**$\pm$0.01
Pythia-160m (layer 8)					
ReLU teacher	122	2.60	76.2	0.404	94.5
RSAE (ReLU-init)	121$\pm$0.1	**1.69**$\pm$0.001	**99.9**$\pm$0.3	**0.187**$\pm$0.002	**97.5**$\pm$0.01
JumpReLU teacher	155	1.34	81.0	0.116	98.4
RSAE (JumpReLU-init)	154$\pm$0.2	**1.31**$\pm$0.001	**100.0**$\pm$0.0	**0.106**$\pm$0.002	**98.5**$\pm$0.01
TopK teacher	160	1.30	99.5	0.125	98.3
RSAE (TopK-init)	160$\pm$0.1	**1.24**$\pm$0.003	**99.9**$\pm$0.0	**0.105**$\pm$0.001	**98.6**$\pm$0.02
Gemma-2-2B (layer 12)					
ReLU teacher	85	2.55	78.6	0.471	95.0
RSAE (ReLU-init)	85$\pm$0.1	**2.15**$\pm$0.001	**97.7**$\pm$0.2	**0.230**$\pm$0.001	**97.6**$\pm$0.01
JumpReLU teacher	84	2.00	99.6	0.177	98.1
RSAE (JumpReLU-init)	83$\pm$0.4	**1.95**$\pm$0.007	**100.0**$\pm$0.0	**0.158**$\pm$0.002	**98.3**$\pm$0.01
TopK teacher	80	2.07	99.9	0.207	97.8
RSAE (TopK-init)	80$\pm$0.4	**2.00**$\pm$0.004	**100.0**$\pm$0.0	**0.197**$\pm$0.001	**97.9**$\pm$0.01

mean number of non-zero latents per token, $\ell_0 = \|\boldsymbol{z}\|_0$; (3) the fraction of alive latents, i.e. latents that fire at least once on the evaluation set; (4) the cross-entropy degradation when the reconstruction replaces the residual stream, $\Delta\text{CE} = \text{CE}_{\hat{\boldsymbol{x}}} - \text{CE}_{\boldsymbol{x}}$ (lower is better); and (5) the loss-recovered fraction $\text{LR} = \frac{\text{CE}_{\text{zero}} - \text{CE}_{\hat{\boldsymbol{x}}}}{\text{CE}_{\text{zero}} - \text{CE}_{\boldsymbol{x}}}$ relative to zero ablation (higher is better). Please refer to Appendix B.1 for the evaluation data and further implementation details.

Choice of degrees. All experiments use a single rational type $(p, q) = (9, 8)$ for every teacher family. The choice is not critical: the identity initialization reproduces ReLU exactly at any degree, and the knee initialization only needs a cubic numerator and a quadratic denominator, so higher degrees add capacity that is used only during fine-tuning, where the learned coefficients beyond degree three remain small. In an ablation on Pythia-160m (ReLU teacher, $d_{\text{sae}} = 2^{12}$, $\ell_0 \approx 160$) the types $(3, 2)$ and $(9, 8)$ ended within 1% of each other on every metric after fine-tuning; $(9, 8)$ is the largest type we found to train stably under the pole-free denominator of (3), and we adopt it throughout so that one architecture covers all teachers. The activation therefore adds only 18 coefficients and two scales to the teacher's parameter count.

Main results. In all eight (model, teacher) pairs of Table 1, the RSAE improves on its teacher in every fidelity metric (MSE, alive, ΔCE, LR) at an ℓ_0 within two latents of the teacher's, and every MSE improvement is at least seven times the across-trial standard deviation. Over all 51 SAEBench teachers on Pythia-160m (Table 5, Appendix B) the RSAE lowers the reconstruction error of 48 and ties three, never increasing it; on MSE, ΔCE and LR it wins 124 of 153 comparisons and loses 3, each by one unit in the last digit.

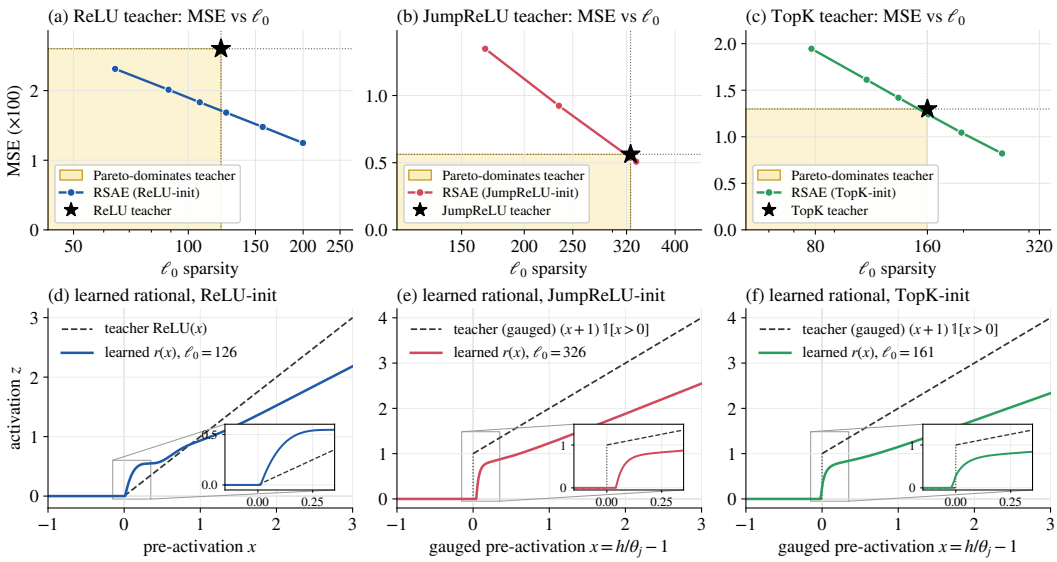


Figure 1: **Pareto fronts and learned activations of RSAE** (Pythia-160m-deduped, layer 8, $d_{\text{sae}} = 2^{12}$, SAEBench core evaluation). **Top (a–c):** reconstruction MSE ($\times 100$) versus ℓ_0 for one SAEBench teacher per family (star) and RSAEs fine-tuned from it at several target sparsities (circles). The shaded box marks the region that Pareto-dominates the teacher, i.e. lower MSE at equal or lower ℓ_0. **Bottom (d–f):** the learned shared rational activation $r(x)$ of type $(9, 8)$ for the RSAE closest to the teacher's ℓ_0, against the teacher's activation (dashed). For JumpReLU and TopK teachers the pre-activation is threshold-gauged, $x = h/\theta_j - 1$, so every teacher latent becomes the unit jump $(x+1)\,\mathbf{1}[x > 0]$. Insets zoom in around $x = 0$. The vertical scale of r trades off against the decoder norms, so only its shape should be compared with the teacher.

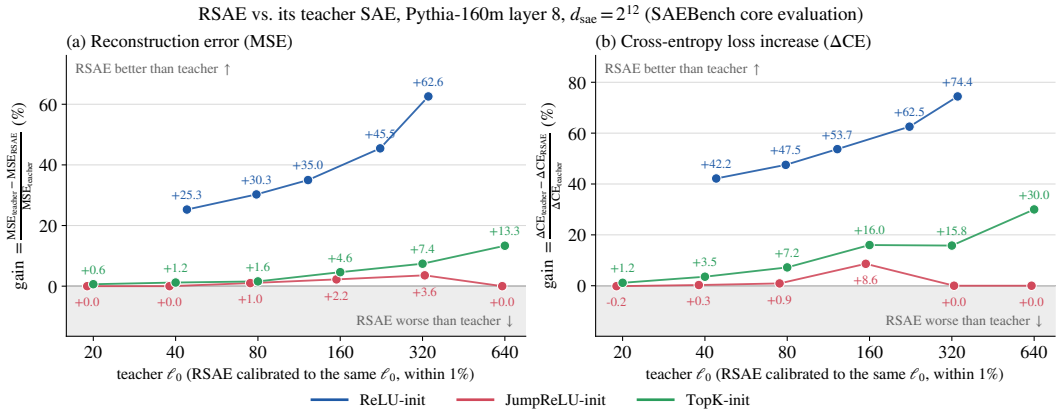


Figure 2: **RSAE improvement over its teacher SAE as a function of sparsity** (Pythia-160m-deduped, layer 8, $d_{\text{sae}} = 2^{12}$, SAEBench core evaluation). The x-axis is the teacher's ℓ_0 on a log scale. **(a)** Relative reduction in reconstruction error, $\frac{\text{MSE}_{\text{teacher}} - \text{MSE}_{\text{RSAE}}}{\text{MSE}_{\text{teacher}}}$. **(b)** Relative reduction in cross-entropy loss increase, $\frac{\Delta\text{CE}_{\text{teacher}} - \Delta\text{CE}_{\text{RSAE}}}{\Delta\text{CE}_{\text{teacher}}}$.

(C1) RSAE improves ReLU SAEs by a large margin. The RSAE reduces the reconstruction error by 11.2%, 35.0% and 15.7% and ΔCE by 31.7%, 53.7% and 51.2% on GPT-2 small, Pythia-160m and Gemma-2-2B. At matched ℓ_0 it closes about 80% of the error gap between the ReLU and the TopK teacher, and in Figure 1(a) it reaches the teacher's error with about half of its active latents. Theorem 1 guarantees only that the RSAE is never worse, since the identity rational reproduces the teacher exactly and fine-tuning starts from it; the size of the gain comes from the response the RSAE learns. Figure 1(d) shows that, starting from the identity, it develops a smooth jump at the origin, the correction that JumpReLU and ProLU hard-code against ℓ_1 shrinkage (Rajamanoharan et al., 2024b;

Table 2: **Isolating the rational activation** (Pythia-160m-deduped, layer 8, $d_{\text{sae}} = 2^{12}$, SAEBench core evaluation). For each teacher, *frozen rational* runs exactly the RSAE pipeline, i.e. the same initialization, data, fine-tuning steps, schedule, auxiliary loss and ℓ_0 calibration, but keeps the rational coefficients and scales $(\boldsymbol{a}, \boldsymbol{b}, C_{\text{in}}, C_{\text{out}})$ fixed at their Step-1 values, so only the encoder and decoder are fine-tuned; for a ReLU teacher this is a ReLU SAE fine-tuned with our objective. All models are calibrated to the teacher's ℓ_0 within 1%. MSE is multiplied by 100; alive and LR are in percent.

Teacher	Model	ℓ_0	MSE↓	alive↑	ΔCE↓	LR↑
ReLU	teacher	122	2.60	76.2	0.404	94.5
	frozen rational	122	2.54	79.7	0.358	94.9
	RSAE	121	**1.69**	**99.9**	**0.187**	**97.5**
JumpReLU	teacher	155	1.34	81.0	0.116	98.4
	frozen rational	155	1.33	99.9	0.118	98.4
	RSAE	154	**1.31**	**100.0**	**0.106**	**98.5**
TopK	teacher	160	1.30	99.5	0.125	98.3
	frozen rational	161	1.29	**99.9**	0.120	98.3
	RSAE	160	**1.24**	**99.9**	**0.105**	**98.6**

Taggart, 2024). The control below confirms that this gain comes from the trainable activation rather than from the additional fine-tuning.

(C2) RSAE reproduces a gate and improves most JumpReLU and TopK SAEs. The knee initialization reproduces a gated teacher to within 1–3% of its error, and the fine-tuned rational keeps a sharp gate at the teacher's gauged threshold (Figure 1(e,f)). The RSAE then lowers MSE by 2.2–11.2% and ΔCE by 4.8–32.4% in all five gated pairs of Table 1, and lowers MSE for 33 of the 36 gated teachers of Table 5, tying the rest. The RSAE follows the structure of the construction behind Theorem 2: its support is a threshold support (the positive set of P), and the threshold re-gauging is the affine reparameterization of Theorem A.7. Gains are an order of magnitude smaller than for ReLU, since gated teachers leave less systematic error. The comparison with the theory is qualitative, as its constants are teacher-dependent.

Isolating the rational activation. Because the RSAE is fine-tuned from its teacher, part of its gain could come from the additional training, the new objective or the auxiliary loss rather than from the rational. Table 2 separates these effects on Pythia-160m by running the identical pipeline with the rational frozen at its Step-1 values. For the ReLU teacher this control, a ReLU SAE fine-tuned with our objective, recovers only a small part of the gain: it lowers MSE from 2.60 to 2.54 and raises the alive fraction from 76.2% to 79.7%, whereas training the rational lowers MSE to 1.69 and raises alive to 99.9%. The trainable activation therefore accounts for 93% of the reconstruction gain, 79% of the ΔCE gain and 85% of the alive-fraction gain. For gated teachers the frozen knee already matches the teacher (MSE 1.33 versus 1.34 for JumpReLU and 1.29 versus 1.30 for TopK), and training the rational adds a further 0.02–0.05 and lowers ΔCE. The rational's contribution beyond reproducing the gate is therefore modest, and the large gains are concentrated on ReLU teachers.

Ablation studies on sparsity and width. Figure 2 and Table 5 show that the gains are consistent across sparsity and width: the RSAE never increases the reconstruction error at any of the 51 (sparsity, width, family) settings, improves MSE, ΔCE and LR for every ReLU teacher, and the ordering ReLU > TopK > JumpReLU holds at every width. The only exception is the alive fraction of ReLU-initialized RSAEs at $d_{\text{sae}} = 2^{16}$, which falls below the teacher's.

Larger budgets k yield larger improvements. Figure 2 shows that the RSAE's advantage grows as the sparsity budget $k = \ell_0$ increases, as anticipated by Theorem 2, where the calibration gain scales as gk while the TopK support-replacement cost grows only as $c\sqrt{k}$. For ReLU teachers the MSE gain rises from 25% at $\ell_0 \approx 40$ to 63% at $\ell_0 \approx 336$ and the ΔCE gain from 42% to 74%; for TopK teachers the MSE gain rises from 0.6% at $\ell_0 \approx 20$ to 13.3% at $\ell_0 \approx 640$ and the ΔCE gain from 1.2% to 30.0%. Table 5 confirms the trend at the larger widths: the ReLU MSE gain rises monotonically at every width, to 75% and 87% at $d_{\text{sae}} = 2^{14}$ and 2^{16}, and the TopK MSE gain rises with a single non-monotone step at each width. JumpReLU follows the trend up to $\ell_0 \approx 320$, from under 1% at $\ell_0 \leq 40$ to 1–5%, and saturates beyond. The growth is in the share of the remaining error that the RSAE removes; in absolute terms the error removed is roughly constant, because denser teachers

leave a smaller residual for the rational to calibrate. At $\ell_0 \approx 640$ the gated teachers leave almost no headroom: all reach 99.9% loss recovered with $\Delta\mathrm{CE} \leq 0.010$, so 12 of the 26 ties in Table 5 occur here and the remaining MSE improvements are at most 0.02×10^{-2}. ReLU teachers, by contrast, remain far from saturation at their densest setting, where the RSAE's gains are largest.

Table 3: **Wall-clock runtime** of the RSAE pipeline per model, averaged across teacher SAEs, measured on a single NVIDIA RTX 5090 (32 GB). The Init Procedure comprises weight inheritance, the closed-form initialization and the distillation of the rational coefficients and scales (skipped for ReLU teachers, whose identity initialization is exact), so its cost varies across teachers; for all three procedures we report mean $\pm$ std across teacher SAEs. Times exclude evaluation and the ℓ_0 calibration rounds, each of which repeats the Fine-tune Procedure (at most three, Appendix B.2).

Model	Init Procedure	Fine-tune Procedure	Total
GPT-2 small	25 ± 11 s	202 ± 120 s	227 ± 120 s
Pythia-160m	199 ± 165 s	1100 ± 210 s	1299 ± 273 s
Gemma-2-2B	197 ± 24 s	1323 ± 290 s	1420 ± 310 s

Runtime and scalability. Table 3 reports the wall-clock cost of the RSAE training procedures and shows two properties of the method: **(i) Cheap approximate teacher reproduction.** With the rational coefficients set in closed form, the only model-specific adaptation of the 18 coefficients and two scales to the teacher pre-activation space completes in a few minutes even on Gemma-2-2B, and brings the RSAE within a few percent of the teacher's reconstruction error before any fine-tuning. **(ii) Lightweight fine-tune overhead.** The rational activation adds only $(p+1)+q+2 = 20$ scalar parameters per RSAE, negligible relative to the $\mathcal{O}(d_{\mathrm{in}} d_{\mathrm{sae}})$ encoder/decoder parameters, and its per-step cost is dominated by producing the language-model activations rather than by the SAE itself. The corresponding 10K-step fine-tune (20.5M tokens, about 4% of the teachers' training budget) costs $\sim$17 min on Pythia-160m and $\sim$22 min on Gemma-2-2B, so a training run takes minutes, not hours, on a single RTX 5090. The ℓ_0 calibration of Appendix B.2 repeats this run at most three more times, so upgrading a released teacher costs between a few minutes and about 1.5 hours.

Sanity check on feature quality. Interpretability is not the target of this work, but we verify that the reconstruction gains do not degrade the learned features: on the eight-task SAEBench sparse-probing panel, the RSAE's full-dictionary probe accuracy on Pythia-160m is on par with its ReLU teacher's (93.71% versus 93.69% on average; Appendix B.3).

6 Conclusion

We introduced the *Rational Sparse Autoencoder* (RSAE), an SAE whose encoder activation is a learnable rational function, and demonstrated its advantages against three popular baseline activation families (ReLU, JumpReLU, TopK) both theoretically and empirically. At matched expected ℓ_0, our theory proves non-inferiority to ReLU and, under stated assumptions, reconstruction advantages over JumpReLU and, for large budgets, fixed raw TopK. On three host language models, we implemented the RSAE as an upgrade of existing pretrained SAEs and observed better fidelity at matched sparsity: the RSAE improves on its teacher in reconstruction error, cross-entropy degradation and loss recovered, and the relative gains grow with the teacher's ℓ_0. Freezing the rational removes most of the gain on ReLU teachers, showing that this gain is due to the trainable activation. Because the upgrade adds only 20 scalar parameters per autoencoder and a training run takes at most about 25 minutes on a single consumer GPU, any released ReLU, JumpReLU, or TopK SAE can in principle be replaced by its RSAE counterpart at small cost.

Limitations. Our evaluation covers three open-weight LLMs up to Gemma-2-2B, with dictionaries of at most 2^{16} latents, and the three most widely released activation families; the gains on larger models and wider dictionaries remain to be measured. Our goal is reconstruction fidelity at fixed sparsity; beyond checking that sparse-probing accuracy is not degraded, we do not study how the RSAE affects feature interpretability. Combining the rational with Gated, BatchTopK or Matryoshka SAEs is left to future work.

AI USE STATEMENT

In this work, we used generative AI tools only for writing and polishing the paper, namely drafting and editing parts of the text to improve readability. We did not use generative AI tools for any of the other tasks listed in the ICLR 2027 AI Policy for Authors. We have reviewed all AI-assisted text, and we take responsibility for the final content of this work, including text and claims produced with the aid of generative AI.

REFERENCES

Bernhard Beckermann and Alex Townsend. On the singular values of matrices with displacement structure. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1227–1248, 2017.

Joseph Bloom. Open source sparse autoencoders for all residual stream layers of GPT2-small. https://www.lesswrong.com/posts/f9EgfLSurAiqRJySD/open-source-sparse-autoencoders-for-all-residual-stream, 2024. LessWrong / HuggingFace release: jbloom/GPT2-Small-SAEs-Reformatted.

Nicolas Boullé, Yuji Nakatsukasa, and Alex Townsend. Rational neural networks. *Advances in neural information processing systems*, 33:14243–14253, 2020.

Dan Braun, Jordan Taylor, Nicholas Goldowsky-Dill, and Lee Sharkey. Identifying functionally important features with end-to-end sparse dictionary learning. *Advances in Neural Information Processing Systems*, 37:107286–107325, 2024.

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

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

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. *arXiv preprint arXiv:2503.17547*, 2025.

Zhiqian Chen, Feng Chen, Rongjie Lai, Xuchao Zhang, and Chang-Tien Lu. Rational neural networks for approximating jump discontinuities of graph convolution operator. *arXiv preprint arXiv:1808.10073*, 2018.

Quentin Delfosse, Patrick Schramowski, Martin Mundt, Alejandro Molina Ramirez, and Kristian Kersting. Adaptive rational activations to boost deep reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 28265–28288, 2024.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, volume 2025, pp. 26721–26754, 2025.

Robert Huben, Hoagy Cunningham, Logan Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations*, volume 2024, pp. 7827–7845, 2024.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Isaac Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum Stuart Mcdougall, Kola Ayonrinde, et al. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability. In *International Conference on Machine Learning*, pp. 29223–29264. PMLR, 2025.

Daniil Laptev, Nikita Balagansky, Yaroslav Aksenov, and Daniil Gavrilov. Analyze feature flow to enhance interpretation and steering in language models. In *International Conference on Machine Learning*, pp. 32593–32616. PMLR, 2025.

Alejandro Molina, Patrick Schramowski, and Kristian Kersting. Padé activation units: End-to-end learning of flexible activation functions in deep networks. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1907.06732.

Donald J Newman. *Approximation with rational functions*. Number 41. American Mathematical Soc., 1979.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, Janos Kramar, Rohin Shah, and Neel Nanda. Improving sparse decomposition of language model activations with gated sparse autoencoders. *Advances in Neural Information Processing Systems*, 37:775–818, 2024a.

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

Glen M. Taggart. ProLU: A nonlinearity for sparse autoencoders. `https://www.lesswrong.com/posts/HEpufTdakGTTKgoYF/prolu-a-pareto-improvement-for-all-sparse-autoencoders`, 2024. LessWrong post.

Maosen Tang and Alex Townsend. Rational neural networks have expressivity advantages. *arXiv preprint arXiv:2602.12390*, 2026.

Matus Telgarsky. Neural networks and rational functions. In *International Conference on Machine Learning*, pp. 3387–3393. PMLR, 2017.

Martin Trimmel, Mihai Zanfir, Richard Hartley, and Cristian Sminchisescu. Era: Enhanced rational activations. In *European Conference on Computer Vision*, pp. 722–738. Springer, 2022.

A Theoretical Derivation

A.1 ReLU Containment

Theorem A.1 (Same-budget non-inferiority to ReLU)**.** *For numerator and denominator degree bounds $p \geq 1$ and $q \geq 0$, every ReLU SAE in* (1) *has a clipped RSAE representation with the same encoder, decoder, biases, and latent width, using one shared rational function and two shared positive scalar scales. The codes and reconstructions agree for every input. Consequently, for any input distribution with finite second moment, the infimum of RSAE reconstruction error at any feasible expected ℓ_0 budget is no greater than the corresponding ReLU infimum under the same encoder and decoder parameter constraints. The same non-inferiority holds for the regularized objective* (6)*, for every $\lambda \geq 0$ and every regularizer that depends only on the code, including ℓ_1.*

Proof. Fix the encoder, decoder, and biases of a ReLU SAE, and write $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{W}_{\text{enc}}(\boldsymbol{x} - \boldsymbol{b}_{\text{dec}}) + \boldsymbol{b}_{\text{enc}}$. Choose

$$P(t) = t, \qquad Q(t) = 1, \qquad r(t) = t, \qquad C_{\text{in}} = C_{\text{out}} = 1.$$

These choices satisfy the degree bounds and give a rational function with no real poles. With $[\cdot]_+$ applied element-wise, the clipped RSAE code obeys

$$\boldsymbol{z}^{\text{Rat}}(\boldsymbol{x}) = C_{\text{out}} \left[r\left(\frac{\boldsymbol{h}(\boldsymbol{x})}{C_{\text{in}}} \right) \right]_+ = [\boldsymbol{h}(\boldsymbol{x})]_+ = \boldsymbol{z}^{\text{ReLU}}(\boldsymbol{x}),$$
$$\widehat{\boldsymbol{x}}^{\text{Rat}}(\boldsymbol{x}) = \boldsymbol{W}_{\text{dec}} \boldsymbol{z}^{\text{Rat}}(\boldsymbol{x}) + \boldsymbol{b}_{\text{dec}} = \widehat{\boldsymbol{x}}^{\text{ReLU}}(\boldsymbol{x}).$$

Thus reconstruction error, ℓ_0 support size, and any regularizer of the code (in particular its ℓ_1 magnitude) are preserved pointwise and hence in expectation. Every feasible ReLU parameter choice is therefore feasible for RSAE at the identical expected ℓ_0 budget with the same objective value. Taking infima proves both non-inferiority statements. □

A.2 General Same-Budget Reconstruction Advantage over Fixed Raw TopK

Under the stability and nonlinear-calibration persistence conditions below, a clipped RSAE code with a shared rational activation attains lower reconstruction risk than fixed raw TopK at the same expected ℓ_0 budget for sufficiently large k. The comparison uses an intermediate threshold support: replacing the fixed top-k support incurs a sublinear cost, whereas nonlinear calibration on the threshold support yields a linear gain.

Consider a sequence of SAE instances indexed by n. For each n, let

$$Y_n \in L^2(\Omega;\mathbb{R}^{d_n}), \qquad D_n = [d_{n,1},\dots,d_{n,M_n}] \in \mathbb{R}^{d_n\times M_n}, \qquad S_n = W_nY_n + b_n,$$

where $W_n \in \mathbb{R}^{M_n\times d_n}$, $b_n \in \mathbb{R}^{M_n}$, and D_n are deterministic and $\|d_{n,j}\|_2 = 1$. Fix $k_n \in \{1,\dots,M_n-1\}$ and assume that the coordinates of S_n have no ties and at least k_n positive entries almost surely. With descending rank, the fixed raw TopK code is

$$z^{\mathrm{T}}_{n,j} = S_{n,j}\mathbf{1}\{\mathrm{rank}_{S_n}(j) \le k_n\}\,.$$

Assume that the pooled score measure $\sum_{j=1}^{M_n}\mathcal{L}(S_{n,j})$ is nonatomic and that there exists $\tau_n > 0$ for which

$$A_{n,j} = \mathbf{1}\{S_{n,j} > \tau_n\}, \qquad N_n = \sum_{j=1}^{M_n} A_{n,j}, \qquad \mathbb{E}N_n = k_n. \tag{7}$$

Define the raw threshold code and reconstruction risk by

$$z^0_{n,j} = S_{n,j}A_{n,j}, \qquad \mathcal{R}_n(z) = \mathbb{E}\|Y_n - D_nz\|_2^2.$$

Thus $\mathbb{E}\|z^{\mathrm{T}}_n\|_0 = k_n = \mathbb{E}\|z^0_n\|_0$. Write $[u]_+ = \max\{u,0\}$.

For each n, let $V_{n,1} > \cdots > V_{n,M_n}$ be the ordered scores and let $\bar{d}_{n,m}$ be the decoder column paired with $V_{n,m}$. Set

$$E_{n,m-1} = Y_n - \sum_{\ell<m}\bar{d}_{n,\ell}V_{n,\ell}, \qquad H_{n,m} = 2V_{n,m}\bar{d}^{\mathsf{T}}_{n,m}E_{n,m-1} - V^2_{n,m}.$$

For the calibration quantities, fix an instance and suppress its index. On the threshold support $A_j = \mathbf{1}\{S_j > \tau\}$, define the raw and shifted active-affine responses

$$a_{\mathrm{J}}(s) = s, \qquad a_{\mathrm{R}}(s) = s - \tau, \qquad s > \tau,$$

and use their affine extensions to $\mathbb{R}$. For $q \in \{\mathrm{J},\mathrm{R}\}$, choose

$$\alpha^{q,*} \in \operatorname*{argmin}_{\alpha\ge 0}\mathbb{E}\left\|Y - \alpha\sum_{j=1}^{M} d_j a_q(S_j)A_j\right\|_2^2, \tag{8}$$

taking the smallest minimizer when it is nonunique. Let (a_*,α^*) be the parameters of the lower-risk fitted response, breaking ties in favor of a_{J}. Define

$$z^{\mathrm{A}}_j = \alpha^* a_*(S_j)A_j, \qquad e_{\mathrm{A}} = Y - Dz^{\mathrm{A}}.$$

To express a shared calibration direction across active coordinates, define the pooled active-coordinate law $\mathbb{P}_*$ by requiring, for every bounded measurable F,

$$\mathbb{E}_*F(J,S_J,Z_J,R_J) = \frac{1}{k}\sum_{j=1}^{M}\mathbb{E}\big[A_jF(j,S_j,z^{\mathrm{A}}_j,d_j^{\mathsf{T}}e_{\mathrm{A}})\big]\,, \qquad Q = Z_JR_J. \tag{9}$$

This is a probability law because $\sum_j \mathbb{E}A_j = k$. For a bounded measurable shared modulation $\phi:\mathbb{R}\to\mathbb{R}$, define

$$X^\phi_j = z^{\mathrm{A}}_j\phi(S_j), \qquad X^\phi = (X^\phi_1,\dots,X^\phi_M)^{\mathsf{T}},$$

For the nth instance, denote the resulting law and expectation by $\mathbb{P}_{*,n}$ and $\mathbb{E}_{*,n}$, and attach the index n likewise to Z_J, Q, and X^ϕ. Set

$$\ell_n(s) = \mathbb{E}_{*,n}[Q_n \mid S_{n,J} = s].$$

Whenever $0 < \mathbb{E}_{*,n}Q_n^2 < \infty$, define its correlation ratio by

$$\eta_n^2 = \frac{\mathbb{E}_{*,n}\ell_n(S_{n,J})^2}{\mathbb{E}_{*,n}Q_n^2}.$$

Assumption A.2 (Stability). *There exist constants $\gamma > 2$, $0 \le \bar{\kappa}_N < \infty$, $0 \le B < \infty$, $0 \le \bar{\kappa}_X < \infty$, and $0 \le A_2 < \infty$, independent of n, and deterministic centers $\lambda_n \in \mathbb{R}$ such that*

$$\begin{aligned} \mathrm{Var}(N_n) &\le \bar{\kappa}_N k_n, \\ \sup_{1 \le m \le M_n} \|H_{n,m} - \lambda_n\|_{L^\gamma} &\le B, \\ \mathbb{E}_{*,n} Z_{n,J}^2 &\le A_2^2. \end{aligned}$$

For every bounded measurable real ϕ with $\|\phi\|_\infty \le 1$,

$$\mathbb{E}\|D_n X_n^\phi\|_2^2 \le \bar{\kappa}_X \sum_{j=1}^{M_n} \mathbb{E}[(X_{n,j}^\phi)^2].$$

Assumption A.3 (Nonlinear-calibration persistence). *There exist constants $\sigma > 0$, $C_Q \in (0, \infty)$, and $\rho \in (0, 1]$, independent of n, such that*

$$\begin{aligned} Q_n \in L^4(\mathbb{P}_{*,n}), \qquad \mathbb{E}_{*,n} Q_n^2 &\ge \sigma^2, \\ \mathbb{E}_{*,n} Q_n^4 &\le C_Q \{\mathbb{E}_{*,n} Q_n^2\}^2, \\ \eta_n^2 &\ge \rho. \end{aligned}$$

Theorem A.4 (General same-budget reconstruction advantage). *Let $k_n \to \infty$, and suppose Assumptions A.2 and A.3 hold for the sequence above. Define*

$$a_0 = \frac{\sigma \rho^{3/2}}{\sqrt{C_Q}}, \qquad q_0 = \bar{\kappa}_X A_2^2, \qquad g_0 = \min\left\{ \frac{a_0}{8}, \frac{a_0^2}{4 q_0} \right\}.$$

There is a constant $c_\gamma < \infty$, depending only on γ, such that, for every n and every $\varepsilon > 0$, there exist a finite-degree shared rational function r with no real poles and a nonconstant reduced denominator, and a shared scalar $C_{\mathrm{out}} > 0$, such that, with $C_{\mathrm{in}} = 1$ in (3), the clipped RSAE code satisfies

$$\begin{aligned} z_{n,j}^{\mathrm{Rat}} &= C_{\mathrm{out}}[r(S_{n,j})]_+, \\ \mathbf{1}\{z_{n,j}^{\mathrm{Rat}} \ne 0\} &= \mathbf{1}\{S_{n,j} > \tau_n\} \quad \textit{almost surely}, \qquad j = 1, \dots, M_n, \\ \mathbb{E}\|z_n^{\mathrm{Rat}}\|_0 &= \mathbb{E} N_n = k_n = \mathbb{E}\|z_n^{\mathrm{T}}\|_0. \end{aligned}$$

The resulting code satisfies

$$\begin{aligned} \mathcal{R}_n(z_n^{\mathrm{Rat}}) - \mathcal{R}_n(z_n^{\mathrm{T}}) &\le \mathcal{R}_n(z_n^0) - \mathcal{R}_n(z_n^{\mathrm{T}}) - g_0 k_n + \varepsilon \\ &\le c_\gamma B \sqrt{\bar{\kappa}_N k_n} - g_0 k_n + \varepsilon. \end{aligned} \tag{10}$$

Hence, for every sufficiently large n, such a code can be chosen to satisfy

$$\mathcal{R}_n(z_n^{\mathrm{Rat}}) < \mathcal{R}_n(z_n^{\mathrm{T}}). \tag{11}$$

The conclusion also applies along sparse sequences satisfying $k_n / M_n \to 0$.

Lemma A.5 (Support-replacement cost). *Fix one instance and suppress its index. For every $\gamma > 2$, there exists $c_\gamma < \infty$, depending only on γ, such that the raw* TopK *and raw threshold codes defined above satisfy, for every $\lambda \in \mathbb{R}$ with $\sup_m \|H_m - \lambda\|_{L^\gamma} < \infty$,*

$$\left| \mathcal{R}(z^{\mathrm{T}}) - \mathcal{R}(z^0) \right| \le c_\gamma \left(\sup_{1 \le m \le M} \|H_m - \lambda\|_{L^\gamma} \right) \sqrt{\mathrm{Var}(N)}. \tag{12}$$

Proof. For $m = 0, \dots, M$, set

$$L_m = \left\| Y - \sum_{\ell \le m} \bar{d}_\ell V_\ell \right\|_2^2. \tag{13}$$

Expansion of the square gives $L_{m-1} - L_m = H_m$. Since the threshold support is the prefix of random length N, the quantity H_m is exactly the reduction in squared reconstruction error from adding rank m to the preceding prefix. Telescoping yields

$$\mathcal{R}(z^{\mathrm{T}}) - \mathcal{R}(z^0) = \mathbb{E} \sum_{m > k} \mathbf{1}\{N \ge m\} H_m - \mathbb{E} \sum_{m \le k} \mathbf{1}\{N < m\} H_m. \tag{14}$$

The identity $\mathbb{E}N = k$ implies

$$\mathbb{E}(N-k)_+ = \mathbb{E}(k-N)_+. \tag{15}$$

Therefore the λ contributions cancel in (14). Set $\vartheta = 1 - 1/\gamma$ and $B_\gamma(\lambda) = \sup_m \|H_m - \lambda\|_{L^\gamma}$. Hölder's inequality then gives

$$\begin{aligned}\left|\mathcal{R}(z^{\mathrm{T}}) - \mathcal{R}(z^0)\right| &\le B_\gamma(\lambda) \sum_{r\ge 1} \left\{\mathbb{P}(N-k \ge r)^\vartheta + \mathbb{P}(k-N \ge r)^\vartheta\right\} \\ &\le 2^{1/\gamma} B_\gamma(\lambda) \sum_{r\ge 1} \mathbb{P}(|N-k| \ge r)^\vartheta. \end{aligned} \tag{16}$$

Let $v = \sqrt{\mathrm{Var}(N)}$. By Chebyshev's inequality,

$$\sum_{r\ge 1} \mathbb{P}(|N-k| \ge r)^\vartheta \le \sum_{r\ge 1} \min\{1, v^2/r^2\}^\vartheta \le C_\gamma v. \tag{17}$$

For $0 < v \le 1$, the last sum is at most $v^{2\vartheta} \sum_{r\ge 1} r^{-2\vartheta} \le v \sum_{r\ge 1} r^{-2\vartheta}$. For $v > 1$, splitting at $\lceil v \rceil$ bounds it by $2v + v/(2\vartheta - 1)$. If $v = 0$, then $N = k$ almost surely and (14) is zero. Taking $c_\gamma = 2^{1/\gamma} C_\gamma$ proves the claim. □

Lemma A.6 (Fixed-support nonlinear calibration gain)**.** *Fix n in a sequence satisfying Assumptions A.2 and A.3, and suppress the instance index. For every $\varepsilon > 0$, there exist a shared scalar $C_{\mathrm{out}} > 0$ and a finite-degree shared rational function r with no real poles and a nonconstant reduced denominator such that*

$$\begin{aligned} z_j^{\mathrm{Rat}} = C_{\mathrm{out}}[r(S_j)]_+, \qquad \mathbf{1}\{z_j^{\mathrm{Rat}} \ne 0\} = \mathbf{1}\{S_j > \tau\} \quad \textit{almost surely}, \\ \mathcal{R}(z^{\mathrm{Rat}}) \le \mathcal{R}(z^{\mathrm{A}}) - g_0 k + \varepsilon \le \mathcal{R}(z^0) - g_0 k + \varepsilon. \end{aligned}$$

Proof. The feasible choice $q = \mathrm{J}$ and $\alpha = 1$ in (8) recovers z^0. Since (a_*, α^*) is the lower-risk solution,

$$\mathcal{R}(z^{\mathrm{A}}) \le \mathcal{R}(z^0). \tag{18}$$

Assumption A.3 implies $\alpha^* > 0$: otherwise $z^{\mathrm{A}} = 0$ and hence $Q = 0$ almost surely. The first-order condition for (8) therefore gives

$$\sum_{j=1}^M \mathbb{E}\left[z_j^{\mathrm{A}} d_j^{\mathsf{T}} e_{\mathrm{A}}\right] = 0. \tag{19}$$

Consequently, $\mathbb{E}_* Q = \mathbb{E}_* \ell(S_J) = 0$ under the pooled law.

We first lower-bound the score-dependent residual alignment available to a shared response. Define finite signed measures

$$\Lambda_j(B) = \mathbb{E}\left[z_j^{\mathrm{A}} d_j^{\mathsf{T}} e_{\mathrm{A}} \mathbf{1}\{S_j \in B\}\right], \qquad \Lambda = \sum_{j=1}^M \Lambda_j, \tag{20}$$

and set

$$\mathsf{S} = \|\Lambda\|_{\mathrm{TV}}, \qquad \mathsf{Z} = \sum_{j=1}^M \mathbb{E}[(z_j^{\mathrm{A}})^2]. \tag{21}$$

Here S measures the shared signal and Z the affine-code energy. The pooled law and conditional expectation give

$$\Lambda(B) = k\mathbb{E}_*[\ell(S_J)\mathbf{1}\{S_J \in B\}], \qquad \mathsf{S} = k\mathbb{E}_*|\ell(S_J)|. \tag{22}$$

Conditional Jensen and Assumption A.3 yield

$$\mathbb{E}_*\ell(S_J)^2 = \eta^2 \mathbb{E}_* Q^2 \ge \rho\sigma^2, \qquad \mathbb{E}_*\ell(S_J)^4 \le \mathbb{E}_* Q^4 \le \frac{C_Q}{\rho^2}\{\mathbb{E}_*\ell(S_J)^2\}^2. \tag{23}$$

Consequently, the L^1–L^2–L^4 interpolation inequality gives

$$\mathbb{E}_*|\ell(S_J)| \ge \frac{\{\mathbb{E}_*\ell(S_J)^2\}^{3/2}}{\{\mathbb{E}_*\ell(S_J)^4\}^{1/2}} \ge \frac{\sigma\rho^{3/2}}{\sqrt{C_Q}} = a_0. \tag{24}$$

Together with Assumption A.2 and (9), this yields

$$\mathsf{S} \geq a_0 k, \qquad \mathsf{Z} = k\mathbb{E}_* Z_J^2 \leq A_2^2 k. \tag{25}$$

We convert this pooled signal into a shared rational direction. Let $g = d\Lambda/d|\Lambda|$ be the polar density. Since $|g| = 1$ almost everywhere with respect to $|\Lambda|$, density of $C_c(\mathbb{R})$ in $L^1(|\Lambda|)$, followed by clipping to $[-1, 1]$, gives $h \in C_c(\mathbb{R})$ such that

$$\|h\|_\infty \leq 1, \qquad \int |h - g| \, d|\Lambda| \leq \frac{\mathsf{S}}{8}.$$

Let $\mathcal{A}$ be the nonunital real algebra generated by $(1 + s^2)^{-1}$ and $s(1 + s^2)^{-1}$. It is a subalgebra of $C_0(\mathbb{R})$ that separates points and vanishes nowhere. The locally compact Stone–Weierstrass theorem therefore makes $\mathcal{A}$ uniformly dense in $C_0(\mathbb{R})$. Choose $p \in \mathcal{A}$ with $\|p - h\|_\infty \leq 1/8$ and set $\phi = (8/9)p$. Then $\|\phi\|_\infty \leq 1$, and

$$\int \phi \, d\Lambda \geq \frac{8}{9}\left(\mathsf{S} - \frac{\mathsf{S}}{8} - \frac{\mathsf{S}}{8}\right) = \frac{2\mathsf{S}}{3}.$$

Every element of $\mathcal{A}$ is a bounded rational function with no real poles. Thus ϕ is rational with no real poles and satisfies

$$\|\phi\|_\infty \leq 1, \qquad \sum_{j=1}^{M} \int \phi \, d\Lambda_j \geq \frac{\mathsf{S}}{2}. \tag{26}$$

For $0 < t < 1/2$, define the shared rational response

$$r_t(s) = a_*(s)\{1 + t\phi(s)\}. \tag{27}$$

Since $1 + t\phi(s) > 0$, the perturbed code $\alpha^* r_t(S_j) A_j$ is positive exactly on the threshold support. Its displacement from z^{A} is

$$\Delta z_j(t) = t X_j^\phi. \tag{28}$$

By (26),

$$\begin{aligned} \mathbb{E}[e_{\mathrm{A}}^{\mathsf{T}} D\Delta z(t)] &= t \sum_{j=1}^{M} \int \phi \, d\Lambda_j \\ &\geq \frac{t\mathsf{S}}{2}. \end{aligned} \tag{29}$$

Assumption A.2 and $|\phi| \leq 1$ bound the quadratic term by

$$\begin{aligned} \mathbb{E}\|D\Delta z(t)\|_2^2 &= t^2 \mathbb{E}\|DX^\phi\|_2^2 \\ &\leq \bar{\kappa}_X \mathsf{Z} t^2. \end{aligned} \tag{30}$$

The exact quadratic expansion of $\mathcal{R}$ gives

$$\mathcal{R}(z^{\mathrm{A}}) - \mathcal{R}(z^{\mathrm{A}} + \Delta z(t)) \geq t\mathsf{S} - t^2\mathsf{C}, \qquad \mathsf{C} = \bar{\kappa}_X \mathsf{Z}. \tag{31}$$

Since $\mathsf{S} > 0$, Equation (29) implies $\mathbb{E}\|DX^\phi\|_2^2 > 0$. Thus $\mathsf{C} > 0$ and $q_0 \geq \mathsf{C}/k > 0$. Choose

$$t_0 = \min\left\{\frac{1}{4}, \frac{\mathsf{S}}{2\mathsf{C}}\right\}. \tag{32}$$

Then

$$\mathcal{R}(z^{\mathrm{A}}) - \mathcal{R}(z^{\mathrm{A}} + \Delta z(t_0)) \geq \min\left\{\frac{\mathsf{S}}{8}, \frac{\mathsf{S}^2}{4\mathsf{C}}\right\}. \tag{33}$$

Equation (25) implies $\mathsf{C} \leq q_0 k$ and $\mathsf{S} \geq a_0 k$; therefore the right-hand side of (33) is at least $g_0 k$.

It remains to realize the perturbed response by a clipped rational function with exactly the threshold support. Define

$$\sigma_\tau(s) = \frac{s - \tau}{1 + (s - \tau)^2}, \qquad f(s) = r_{t_0}(s)\mathbf{1}\{s > \tau\}, \qquad \nu = \sum_{j=1}^{M} \mathcal{L}(S_j). \tag{34}$$

Since $a_*(s) > 0$ and $1 + t_0\phi(s) > 0$ on (τ, ∞), the function f is nonnegative. Moreover, $S_j \in L^2$ and the boundedness of ϕ imply $f \in L^2(\nu)$. Define $u(s) = \sqrt{f(s)/\sigma_\tau(s)}$ for $s > \tau$ and $u(s) = 0$ otherwise. Let ζ be the measure defined by $d\zeta = \sigma_\tau^2 d\nu$. Then

$$\|u\|_{L^4(\zeta)}^4 = \int_{(\tau,\infty)} f(s)^2 \, d\nu(s) < \infty. \tag{35}$$

First approximate u in $L^4(\zeta)$ by functions in $C_c(\mathbb{R})$, which is dense in that space. Uniform approximation by the rational algebra above then yields finite-degree rational functions q_m with no real poles such that $q_m \to u$ in $L^4(\zeta)$. For any positive sequence $\xi_m \downarrow 0$, set

$$\widetilde{r}_m(s) = \sigma_\tau(s) \left\{ q_m(s)^2 + \frac{\xi_m}{1+s^2} \right\}. \tag{36}$$

The term in braces is strictly positive, and hence

$$\{s : \widetilde{r}_m(s) > 0\} = (\tau, \infty). \tag{37}$$

Moreover,

$$\|q_m^2 - u^2\|_{L^2(\zeta)} \le \|q_m - u\|_{L^4(\zeta)} \big(\|q_m\|_{L^4(\zeta)} + \|u\|_{L^4(\zeta)} \big) \longrightarrow 0. \tag{38}$$

The sign identity above and the triangle inequality now give

$$\begin{aligned} \|[\widetilde{r}_m]_+ - f\|_{L^2(\nu)} &\le \|q_m^2 - u^2\|_{L^2(\zeta)} \\ &\quad + \xi_m \left\| \mathbf{1}_{(\tau,\infty)} \frac{\sigma_\tau(s)}{1+s^2} \right\|_{L^2(\nu)} \longrightarrow 0. \end{aligned}$$

The second norm is finite because $\nu(\mathbb{R}) = M$ and its integrand is bounded. Each $\widetilde{r}_m$ is bounded, has no real poles, and changes sign at τ. Moreover, its reduced denominator is nonconstant, since a bounded polynomial is constant whereas $\widetilde{r}_m$ changes sign.

Define

$$z_j^{(m)} = \alpha^* [\widetilde{r}_m(S_j)]_+. \tag{39}$$

The shared scale α^* is strictly positive, the support is exactly $\{j : S_j > \tau\}$, and $Dz^{(m)}$ converges in L^2 to $D(z^{\mathrm{A}} + \Delta z(t_0))$. Continuity of the squared L^2 risk and (33) give, for all sufficiently large m,

$$\mathcal{R}(z^{(m)}) \le \mathcal{R}(z^{\mathrm{A}}) - g_0 k + \varepsilon. \tag{40}$$

Choosing such an m and setting $r = \widetilde{r}_m$, $C_{\mathrm{out}} = \alpha^*$, and $z^{\mathrm{Rat}} = z^{(m)}$ proves the claim. □

Proof of Theorem A.4. Lemma A.5 and Assumption A.2 give

$$\big| \mathcal{R}_n(z_n^{\mathrm{T}}) - \mathcal{R}_n(z_n^0) \big| \le c_\gamma B \sqrt{\bar{\kappa}_N k_n}. \tag{41}$$

Lemma A.6 yields

$$\begin{aligned} \mathcal{R}_n(z_n^{\mathrm{Rat}}) - \mathcal{R}_n(z_n^{\mathrm{T}}) &\le \mathcal{R}_n(z_n^0) - \mathcal{R}_n(z_n^{\mathrm{T}}) - g_0 k_n + \varepsilon \\ &\le c_\gamma B \sqrt{\bar{\kappa}_N k_n} - g_0 k_n + \varepsilon, \end{aligned} \tag{42}$$

which proves (10). The construction and support identity in Lemma A.6 give the displayed code and exact support claims, and (7) gives the matched expected ℓ_0 budget. Since $g_0 > 0$ and $k_n \to \infty$, choose

$$0 < \varepsilon_n < g_0 k_n - c_\gamma B \sqrt{\bar{\kappa}_N k_n} \tag{43}$$

for every sufficiently large n. Substitution into (10) proves (11). □

A.3 General Same-Budget Reconstruction Advantage over JumpReLU

The nonlinear-calibration argument also applies to JumpReLU with per-feature thresholds. Let $Y \in L^2(\Omega; \mathbb{R}^d)$, $S = WY + b \in \mathbb{R}^M$, and let $D = [d_1, \ldots, d_M]$ have unit-norm columns. For thresholds $\theta_j > 0$, define

$$A_j = \mathbf{1}\{S_j > \theta_j\}, \qquad z_j^{\mathrm{J}} = S_j A_j, \qquad k = \sum_{j=1}^M \mathbb{E} A_j > 0, \qquad \mathcal{R}(z) = \mathbb{E}\|Y - Dz\|_2^2.$$

Fit a shared scalar by

$$\alpha^* \in \operatorname*{argmin}_{\alpha \geq 0} \mathcal{R}(\alpha z^{\mathrm{J}}), \qquad z^{\mathrm{A}} = \alpha^* z^{\mathrm{J}}, \qquad e_{\mathrm{A}} = Y - D z^{\mathrm{A}},$$

taking the smallest minimizer when it is nonunique. Define the pooled law $\mathbb{P}_*$ by (9) with these A_j, z^{A}, and k, and define Q, ℓ, η, and X^ϕ as in Appendix A.2.

Theorem A.7 (General reconstruction advantage over JumpReLU)**.** *Suppose the bounds on $\mathbb{E}_* Z_J^2$ and $\mathbb{E}\|DX^\phi\|_2^2$ in Assumption A.2, and Assumption A.3, hold for the quantities above. Then $q_0 = \bar{\kappa}_X A_2^2 > 0$. Define g_0 from these constants as in Theorem A.4. For every $\varepsilon > 0$, there exist an affine encoder $\widehat{S} = \widehat{W} Y + \widehat{b}$, a finite-degree shared rational function r with no real poles and a nonconstant reduced denominator, and a shared scalar $C_{\mathrm{out}} > 0$, such that, with $C_{\mathrm{in}} = 1$, the clipped RSAE code satisfies*

$$\begin{aligned} z_j^{\mathrm{Rat}} &= C_{\mathrm{out}} [r(\widehat{S}_j)]_+, \qquad \mathbf{1}\{z_j^{\mathrm{Rat}} \neq 0\} = A_j \quad \textit{almost surely}, \\ \mathbb{E}\|z^{\mathrm{Rat}}\|_0 &= k = \mathbb{E}\|z^{\mathrm{J}}\|_0, \\ \mathcal{R}(z^{\mathrm{Rat}}) &\leq \mathcal{R}(z^{\mathrm{J}}) - g_0 k + \varepsilon. \end{aligned}$$

The decoder D and latent width M are unchanged, and $g_0 > 0$.

Proof. The feasible choice $\alpha = 1$ gives $\mathcal{R}(z^{\mathrm{A}}) \leq \mathcal{R}(z^{\mathrm{J}})$. Persistence implies $\alpha^* > 0$, and scalar stationarity gives $\mathbb{E}_* Q = 0$. The signed-measure argument in (20)–(25) therefore gives a pooled calibration signal of at least $a_0 k$. Construction (26) yields a bounded pole-free rational ϕ with $\|\phi\|_\infty \leq 1$; the decoded-modulation bound then gives the linear gain and quadratic cost in (31). The positive residual alignment forces $\mathbb{E}\|DX^\phi\|_2^2 > 0$ by Cauchy–Schwarz, hence $\bar{\kappa}_X > 0$. Persistence also gives $\mathbb{E}_* Z_J^2 > 0$, so $A_2 > 0$ and $q_0 > 0$. The step size $t_0 \in (0, 1/4]$ in (32) gives

$$f(s) = s\{1 + t_0 \phi(s)\}, \qquad \bar{z}_j = \alpha^* f(S_j) A_j, \qquad \mathcal{R}(\bar{z}) \leq \mathcal{R}(z^{\mathrm{A}}) - g_0 k.$$

In particular, $0 < f(s) \leq 2s$ for $s > 0$.

We realize this calibrated code using a shared rational response and an affine encoder. Fix $\delta_0 > 0$. For $R > \max_j \theta_j$ and $0 < \delta < \min_j (R - \theta_j)$, choose $c_1, \ldots, c_M > 0$ recursively so that

$$c_{j+1}\delta > c_j (R - \theta_j), \qquad j = 1, \ldots, M-1.$$

The intervals $I_j = [c_j \delta, c_j (R - \theta_j)]$, $j = 1, \ldots, M$, are disjoint. On their union, set

$$F(u) = f(\theta_j + u/c_j), \qquad u \in I_j,$$

and extend F continuously between the intervals and to zero outside a compact subset of $(0, \infty)$, with $0 \leq F \leq 2R$. Define the affine encoder by

$$\widehat{W} = \operatorname{diag}(c_1, \ldots, c_M) W, \qquad \widehat{b} = \operatorname{diag}(c_1, \ldots, c_M)(b - \theta), \qquad \widehat{S}_j = c_j (S_j - \theta_j),$$

where $\theta = (\theta_1, \ldots, \theta_M)^{\mathsf{T}}$. For $\theta_j + \delta \leq S_j \leq R$, $F(\widehat{S}_j) = f(S_j)$, while both $F(\widehat{S}_j)$ and $f(S_j) A_j$ vanish when $S_j \leq \theta_j$. Using $0 \leq F \leq 2R$ and $0 < f(s) \leq 2s$ for $s > 0$ on the remaining events gives

$$\begin{aligned} \sum_{j=1}^M \mathbb{E}|F(\widehat{S}_j) - f(S_j) A_j|^2 &\leq 16 \sum_{j=1}^M \mathbb{E}[S_j^2 \mathbf{1}\{S_j > R\}] \\ &\quad + 16 R^2 \sum_{j=1}^M \mathbb{P}(\theta_j < S_j < \theta_j + \delta). \end{aligned}$$

Since $S_j \in L^2$, the first term tends to zero as $R \to \infty$. For each fixed R, the second tends to zero as $\delta \downarrow 0$ by continuity of probability for decreasing events. Choose R and then δ so that the displayed bound is less than $\delta_0^2/4$.

Set $\sigma_0(u) = u/(1 + u^2)$ and define $v(u) = \sqrt{F(u)/\sigma_0(u)}$ for $u > 0$, with $v(u) = 0$ for $u \leq 0$. Because F is continuous with compact support in $(0, \infty)$, $v \in C_c(\mathbb{R})$. Uniform approximation in

the rational algebra used in Lemma A.6 gives bounded pole-free rational functions q_m with $q_m \to v$ uniformly. For $\xi_m > 0$ tending to zero, define

$$r_m(u) = \sigma_0(u)\left\{q_m(u)^2 + \frac{\xi_m}{1+u^2}\right\}.$$

These functions are bounded and have no real poles. The expression in braces is strictly positive, so $\{u : r_m(u) > 0\} = (0, \infty)$. Their reduced denominators are nonconstant: a bounded polynomial is constant, whereas r_m changes sign at zero. Moreover,

$$\|[r_m]_+ - F\|_\infty \le \frac{1}{2}\|q_m - v\|_\infty(\|q_m\|_\infty + \|v\|_\infty) + \frac{\xi_m}{2} \longrightarrow 0.$$

Choose m so that this norm is less than $\delta_0/(2\sqrt{M})$. The triangle inequality in $L^2(\Omega; \mathbb{R}^M)$ gives

$$\left(\sum_{j=1}^{M} \mathbb{E}|[r_m(\widehat{S}_j)]_+ - f(S_j)A_j|^2\right)^{1/2} < \delta_0.$$

Set $C_{\text{out}} = \alpha^*$ and $z_j^{\text{Rat}} = \alpha^*[r_m(\widehat{S}_j)]_+$. Since $c_j > 0$, its support is exactly $\{j : S_j > \theta_j\}$. The decoded approximation error satisfies

$$\|D(z^{\text{Rat}} - \bar{z})\|_{L^2} \le \alpha^* \|D\|_{\text{op}} \delta_0.$$

Choosing δ_0 sufficiently small makes $\mathcal{R}(z^{\text{Rat}}) \le \mathcal{R}(\bar{z}) + \varepsilon$ by continuity of the squared L^2 risk. Combining the risk inequalities and the support identity proves the theorem. □

Theorem 2 (restated). *Under Assumptions 1 and 2, in the settings of Theorems A.4 and A.7, there are constants $c \ge 0$ and $g > 0$, depending only on the assumption bounds, such that for every $\varepsilon > 0$ there exists an RSAE with a finite-degree shared rational activation without real poles, the same decoder and latent width, and $\mathbb{E}\|z^{\text{Rat}}\|_0 = k$, satisfying*

$$\mathcal{R}(z^{\text{base}}) - \mathcal{R}(z^{\text{Rat}}) \ge \begin{cases} gk - c\sqrt{k} - \varepsilon, & \text{base} = \text{TopK}, \\ gk - \varepsilon, & \text{base} = \text{JumpReLU}. \end{cases}$$

The encoder is unchanged for TopK; an affine reparameterization preserves the per-feature support for JumpReLU.

Proof. For TopK, Theorem A.4 gives the stated inequality with $c = c_\gamma B\sqrt{\bar{\kappa}_N}$ and $g = g_0$. For JumpReLU, Theorem A.7 gives the stated inequality with $g = g_0$, using its corresponding assumption constants. Both constructions preserve the decoder, latent width, and expected support size, and use a finite-degree shared rational activation without real poles. The TopK construction leaves the encoder unchanged; the JumpReLU construction uses an affine reparameterization preserving each feature's support. □

B Detailed Empirical Results

B.1 Experimental Setup

Models and hook points. We evaluate on three open-weight language models: **GPT-2 small** at layer 6 ($d_{\text{in}} = 768$), **Pythia-160m-deduped** at layer 8 (`blocks.8.hook_resid_post`, $d_{\text{in}} = 768$), and **Gemma-2-2B** at layer 12 (`blocks.12.hook_resid_post`, $d_{\text{in}} = 2304$). For GPT-2 small each teacher is used at its own hook point: `blocks.6.hook_resid_pre` for the ReLU teacher and `blocks.6.hook_resid_post` for the TopK teacher. Gemma-2-2B is run in `bf16`, as in SAEBench; GPT-2 small and Pythia-160m are run in `fp32`.

Teacher SAE checkpoints. Teacher SAEs are pre-trained, publicly released checkpoints. For GPT-2 small the ReLU teacher is `gpt2-small-res-jb` (Bloom, 2024) ($d_{\text{sae}} = 24{,}576$) and the TopK teacher is the OpenAI-v5 `gpt2-small-resid-post-v5-32k` (Gao et al., 2025) ($d_{\text{sae}} = 32{,}768$, with layer-norm preprocessing of the residual stream). For

Algorithm 1 Rational Sparse Autoencoder (RSAE) construction.

Require: Pre-trained teacher SAE $\{\widetilde{\boldsymbol{W}}_{\text{enc}}, \widetilde{\boldsymbol{b}}_{\text{enc}}, \widetilde{\boldsymbol{W}}_{\text{dec}}, \widetilde{\boldsymbol{b}}_{\text{dec}}\}$ with activation $\phi^{\text{teacher}} \in \{\text{ReLU}, \text{JumpReLU}_{\tilde{\theta}}, \text{TopK}_k\}$; rational type (p, q); calibration batch $\{\boldsymbol{x}_n\}_{n=1}^{N_{\text{cal}}}$; sparsity regularizer S with coefficients $\mu \geq 0$, $\lambda > 0$; knee sharpness K.

Step 1 — RSAE Initialization Procedure.

if ϕ^{teacher} is JumpReLU or TopK **then**

  Absorb the thresholds into the teacher weights: $\widetilde{\boldsymbol{W}}_{\text{enc}}[:, j] \leftarrow \widetilde{\boldsymbol{W}}_{\text{enc}}[:, j]/\tilde{\theta}_j$, $\widetilde{b}_{\text{enc},j} \leftarrow \widetilde{b}_{\text{enc},j}/\tilde{\theta}_j - 1$, $\widetilde{\boldsymbol{W}}_{\text{dec}}[j, :] \leftarrow \tilde{\theta}_j \, \widetilde{\boldsymbol{W}}_{\text{dec}}[j, :]$ ▷ every feature becomes $(v+1)\,\mathbf{1}[v > 0]$

end if

Initialize the encoder and decoder with $\{\widetilde{\boldsymbol{W}}_{\text{enc}}, \widetilde{\boldsymbol{b}}_{\text{enc}}, \widetilde{\boldsymbol{W}}_{\text{dec}}, \widetilde{\boldsymbol{b}}_{\text{dec}}\}$ and compute the teacher pre-activations $\boldsymbol{h}_n$ on the calibration batch.

Set $C_{\text{in}} = C_{\text{out}} = \text{std}(\boldsymbol{h})$.

Identity-init: $\boldsymbol{a}^* = (0, 1, 0, \ldots, 0)$, $\boldsymbol{b}^* = (1, 0, \ldots, 0)$. ▷ exactly ReLU after the clamp

if ϕ^{teacher} is JumpReLU or TopK **then**

  Knee-init: $r_{(\boldsymbol{a}^*, \boldsymbol{b}^*)}(t) = \dfrac{a_1 t + a_2 t^2 + a_3 t^3}{1 + K^2 t^2}$ with $a_3 = K^2 \frac{C_{\text{in}}}{C_{\text{out}}}$, $a_2 = \frac{K^2}{C_{\text{out}}}$, $a_1 = \max\!\left(\frac{K}{C_{\text{out}}}, 1.05\,\frac{a_2^2}{4a_3}\right)$.

end if

for each initialization of $(\boldsymbol{a}, \boldsymbol{b})$ **do** ▷ identity only for ReLU; knee and identity for gated

  With the encoder and decoder frozen, minimize (5) over $(\boldsymbol{a}, \boldsymbol{b}, C_{\text{in}}, C_{\text{out}})$.

end for

Keep the best fit $(\widetilde{\boldsymbol{a}}, \widetilde{\boldsymbol{b}}, \widetilde{C}_{\text{in}}, \widetilde{C}_{\text{out}})$.

Step 2 — RSAE Fine-Tuning Procedure.

Initialize the encoder and decoder with the (rescaled) teacher weights and the rational with $(\widetilde{\boldsymbol{a}}, \widetilde{\boldsymbol{b}}, \widetilde{C}_{\text{in}}, \widetilde{C}_{\text{out}})$; unfreeze $\Theta := \{\boldsymbol{W}_{\text{enc}}, \boldsymbol{b}_{\text{enc}}, \boldsymbol{W}_{\text{dec}}, \boldsymbol{b}_{\text{dec}}, \boldsymbol{a}, \boldsymbol{b}, C_{\text{in}}, C_{\text{out}}\}$.

repeat

  Sample a mini-batch $\boldsymbol{x} \sim \mathcal{D}$; compute $\boldsymbol{h} = \boldsymbol{W}_{\text{enc}}(\boldsymbol{x} - \boldsymbol{b}_{\text{dec}}) + \boldsymbol{b}_{\text{enc}}$, $\boldsymbol{z} = \phi(\boldsymbol{h})$ via (3), and $\hat{\boldsymbol{x}} = \boldsymbol{W}_{\text{dec}}\boldsymbol{z} + \boldsymbol{b}_{\text{dec}}$.

  Take an Adam step on Θ for (6), $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 + \lambda\, S(\boldsymbol{z})$. ▷ details in Appendix B.2

until the step budget is reached

Ensure: RSAE with parameters Θ.

Pythia-160m, the ReLU, JumpReLU and TopK teachers are taken from the SAEBench (Karvonen et al., 2025) releases `adamkarvonen/saebench_pythia-160m-deduped_width-2pow{12,14,16}_date-0108`, which provide six trainers (sparsity levels) per family at each width $d_{\text{sae}} \in \{2^{12}, 2^{14}, 2^{16}\}$; the main results use $d_{\text{sae}} = 2^{12}$. For Gemma-2-2B all three teacher families are taken from the SAEBench release `adamkarvonen/saebench_gemma-2-2b_width-2pow12_date-0108` ($d_{\text{sae}} = 4{,}096$, the only released width).

Activation buffer and calibration. Training activations are streamed through TransformerLens hooks. For Pythia-160m and Gemma-2-2B they come from the Pile (`monology/pile-uncopyrighted`) and reproduce the SAEBench activation buffer the teachers were trained with (contexts of 1024 tokens with the first token dropped, batches of 2048 tokens drawn from a shuffled buffer); for GPT-2 small they come from OpenWebText. The first $4{,}096$ tokens of the stream form the calibration buffer: it sets the shared input and output scales $C_{\text{in}} = C_{\text{out}}$ in (3) to the standard deviation of the teacher's pre-activations, and the activation distillation step (5) is fitted on it (on a $2{,}048$-token subsample for gated teachers). The fine-tuning procedure then trains on $10{,}000$ fresh batches (≈ 20.5M tokens), with inputs divided by the dataset norm factor as in SAEBench.

Hardware. All RSAE training reported in Table 3 is run on a single NVIDIA RTX 5090 (32 GB). End-to-end wall-clock per (model, teacher) is dominated by the fine-tuning procedure. Runs at $d_{\text{sae}} = 2^{12}$ on Pythia-160m were executed two at a time on the same GPU, so their wall-clock times are upper bounds.

Evaluation protocol. All metrics are computed with the SAEBench core evaluation (Karvonen et al., 2025), applied identically to every teacher and every RSAE, on OpenWebText documents packed into 128-token windows that each start with a BOS token (special-token positions are excluded from the averages). For Pythia-160m and Gemma-2-2B this evaluation text is disjoint from the training text. Reconstruction MSE (squared error summed over d_{in}, averaged over tokens), ℓ_0 (the number of exactly non-zero latents per token) and the alive fraction (latents with non-zero activation density) are computed on $4{,}096{,}000$ tokens. Cross-entropy degradation $\Delta\text{CE} = \text{CE}_{\text{SAE}} - \text{CE}_{\text{clean}}$ and the loss-recovered fraction $\text{LR} = (\text{CE}_{\text{zero}} - \text{CE}_{\text{SAE}})/(\text{CE}_{\text{zero}} - \text{CE}_{\text{clean}})$ are computed on $409{,}600$ tokens ($3{,}200$ windows), by replacing the residual stream at the SAE's host layer with $\hat{\boldsymbol{x}}$ and reading the language model's next-token cross-entropy at every position; CE_{zero} is the cross-entropy obtained when the residual stream at the host layer is replaced by the zero vector.

Sparse-probing setup (Table 4). We use the SAEBench probing suite of 8 binary-classification tasks. Per dataset we train an ℓ_2-regularized logistic regression on the full SAE feature vector (no k-sparsity constraint) using SAEBench's default split, and report mean test accuracy across the dataset's evaluation seeds.

B.2 Implementation Details of the RSAE Initialization and Training

Threshold re-gauging. A JumpReLU teacher computes $\phi_j^{\text{teacher}}(\boldsymbol{h}) = h_j\,\mathbf{1}[h_j > \tilde{\theta}_j]$ with a learned per-feature threshold $\tilde{\theta}_j > 0$; for a TopK teacher $\tilde{\theta}_j \equiv \tilde{\theta}$ is the released inference threshold (main text, footnote). Since the RSAE activation is shared across features, it cannot express d_{sae} different jump heights, so feature j is rescaled by $\tilde{\theta}_j$ before initialization: $\widetilde{\boldsymbol{W}}_{\text{enc}}[:,j] \leftarrow \widetilde{\boldsymbol{W}}_{\text{enc}}[:,j]/\tilde{\theta}_j$, $\widetilde{b}_{\text{enc},j} \leftarrow \widetilde{b}_{\text{enc},j}/\tilde{\theta}_j - 1$, $\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:] \leftarrow \tilde{\theta}_j\,\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:]$. Writing $v_j = (h_j - \tilde{\theta}_j)/\tilde{\theta}_j$ for the rescaled pre-activation, the teacher's contribution to the reconstruction is unchanged, $h_j\mathbf{1}[h_j > \tilde{\theta}_j]\,\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:] = \big[(v_j+1)\mathbf{1}[v_j > 0]\big]\big(\tilde{\theta}_j\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:]\big)$, and every feature now computes the same unit jump $\varphi(v) = (v+1)\,\mathbf{1}[v>0]$. Thresholds are floored at $0.1 \cdot \text{median}_j\tilde{\theta}_j$ so that dead features with $\tilde{\theta}_j \approx 0$ do not blow up the encoder scale. After the rescaling the decoder rows are no longer unit-norm, so for re-gauged teachers the decoder is not re-normalized during fine-tuning. Because the threshold-gated TopK teacher matches the exact top-k teacher in mean ℓ_0 but not token by token, the Step 1 gap is slightly larger for TopK than for JumpReLU teachers.

Knee coefficients. With $t = v/C_{\text{in}}$, the tail of $r(t) = (a_1t + a_2t^2 + a_3t^3)/(1 + K^2t^2)$ is $r(t) = \frac{a_3}{K^2}t + \frac{a_2}{K^2} + O(1/t)$, so condition (i) of the main text, $C_{\text{out}}\,r(v/C_{\text{in}}) = v + 1 + O(1/v)$, gives $a_3 = K^2C_{\text{in}}/C_{\text{out}}$ and $a_2 = K^2/C_{\text{out}}$. Near the origin $r(t) \approx a_1t$ reaches the tail intercept $1/C_{\text{out}}$ at $t \approx 1/(a_1C_{\text{out}})$, so condition (ii), a knee of width $\approx 1/K$, is $a_1 \geq K/C_{\text{out}}$. For condition (iii) note that $r(t) = t\,(a_1 + a_2t + a_3t^2)/(1 + K^2t^2)$ with a positive denominator, so $r(t) < 0$ for all $t < 0$ iff the quadratic has no real root, $a_2^2 < 4a_1a_3$; $a_1 = \max(K/C_{\text{out}},\, 1.05\,a_2^2/(4a_3))$ satisfies both with a 5% margin. In the notation of (3), $\boldsymbol{a}^* = (0, a_1, a_2, a_3, 0, \ldots, 0)$ and $\boldsymbol{b}^* = (1, 0, K^2, 0, \ldots, 0)$, for which the safe denominator reduces to $1 + K^2t^2$. Sharper knees ($K \geq 120$) fit the jump more closely after distillation but drift during fine-tuning (the reconstruction error rises during the learning-rate warm-up and does not recover) and were not used. Because the knee is not optimal for very sparse teachers, Step 1 runs the distillation (5) from both the identity and the knee initialization and keeps the lower objective; the identity start wins at $\ell_0 \approx 20$ and the knee everywhere else.

Architecture. All experiments use a shared rational of type $(p, q) = (9, 8)$ with the safe denominator of (3), so the activation adds 18 coefficients and two scales to the teacher's parameter count. The scales are parameterized as $C_{\text{in}} = e^{\gamma_{\text{in}}}$, $C_{\text{out}} = e^{\gamma_{\text{out}}}$ with γ trainable. The encoder input convention follows the teacher: SAEBench Standard and TopK SAEs subtract $\widetilde{\boldsymbol{b}}_{\text{dec}}$ before the encoder, the SAEBench JumpReLU SAEs do not.

Decoder-norm folding. SAEBench Standard (ReLU) SAEs are released with non-unit decoder rows (row norms 0.45–0.74 for Pythia-160m). Since fine-tuning re-normalizes $\boldsymbol{W}_{\text{dec}}$ after every step, we first apply the exact reparameterization $\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:] \leftarrow \widetilde{\boldsymbol{W}}_{\text{dec}}[j,:]/n_j$, $\widetilde{\boldsymbol{W}}_{\text{enc}}[:,j] \leftarrow n_j\widetilde{\boldsymbol{W}}_{\text{enc}}[:,j]$, $\widetilde{b}_{\text{enc},j} \leftarrow n_j\widetilde{b}_{\text{enc},j}$ with $n_j = \|\widetilde{\boldsymbol{W}}_{\text{dec}}[j,:]\|_2$, which is valid for any positively homogeneous activation (ReLU and the identity-initialized rational). Without it the first fine-tuning step moves the model

far from the teacher (reconstruction error $0.058 \to 0.168$ on the Pythia-160m ReLU teacher with $\ell_0 \approx 160$ at $d_{\text{sae}} = 2^{14}$).

Sparsity regularizer. We use a target-ℓ_0 regularizer $S(\boldsymbol{z}) = \big(\mathbb{E}[\widehat{\ell}_0]/(\rho\,\ell_0^{\text{teacher}}) - 1\big)^2$, where ℓ_0^{teacher} is the teacher's mean number of active features, ρ a target fraction and $\widehat{\ell}_0$ a differentiable count of the RSAE's support $\{j : P(h_j/C_{\text{in}}) > 0\}$; no threshold parameter exists, so S acts on sparsity only through the rational coefficients, the scales and the encoder. For ReLU teachers, whose rational is close to linear at its root, we count on the activation, $\widehat{\ell}_0 = \sum_j \sigma\big(C_{\text{out}}\, r_{(\boldsymbol{a},\boldsymbol{b})}(h_j/C_{\text{in}})/\varepsilon\big)$ with σ the logistic function and $\varepsilon = 10^{-3}$ in normalized activation units. For gated teachers the rational is a steep knee whose inactive side stays within ε of zero over a wide range of h, which inflates this count (soft 190–209 for a hard ℓ_0 of 160) and over-sparsifies the model early in training; we therefore count in pre-activation space, $\widehat{\ell}_0 = \sum_j \sigma\big((h_j - \theta_{\boldsymbol{h}})/\varepsilon\big)$ with $\theta_{\boldsymbol{h}} = C_{\text{in}} u^*$ and u^* the first upward zero crossing of P on $[-4, 4]$, located by bisection and differentiated through the implicit function theorem, $\partial u^*/\partial a_k = -(u^*)^k/P'(u^*)$, implemented as one Newton correction with automatic differentiation. This form is complemented by the penalty $\varepsilon^{-1}\sum_j [C_{\text{out}} r(h_j/C_{\text{in}})]_+ \mathbf{1}[h_j < \theta_{\boldsymbol{h}} - 4\varepsilon]$, which forbids a second positive lobe of P far from the root (without it one run activated almost every feature on strongly negative pre-activations while the soft count still read the target). In (5) the activation-space count with $\rho = 1$ and $\mu = 1$ is used; in (6) $\lambda = 1$. An ℓ_1 penalty, $S(\boldsymbol{z}) = \|\boldsymbol{z}\|_1$, recovers the classical objective but, for any coefficient, only retraces the teacher's Pareto front in our experiments.

Calibration of ρ. The evaluated ℓ_0 of a trained RSAE lands close to, but not exactly at, $\rho\,\ell_0^{\text{teacher}}$. We therefore treat ρ as the single tuned hyper-parameter: starting from $\rho = 1.02$ (TopK) or 1.00 (JumpReLU, ReLU), we train, evaluate ℓ_0 under the protocol below, and if $|\ell_0/\ell_0^{\text{teacher}} - 1| > 0.01$ retrain with $\rho \leftarrow \rho\,\ell_0^{\text{teacher}}/\ell_0$, for at most three rounds with $\rho \in [0.85, 1.15]$ and degenerate runs ($\ell_0/\ell_0^{\text{teacher}} \notin [0.5, 1.5]$) ignored. All reported RSAEs use at most 1% more active latents than their teacher, and most are within 1% of the teacher's ℓ_0. The exceptions are all sparser than their teacher: by up to 5% for ReLU teachers at $d_{\text{sae}} \in \{2^{14}, 2^{16}\}$ and for the GPT-2 small teachers, and by one latent for the TopK teachers at $\ell_0 \approx 80$ (Tables 1 and 5).

Step 1 distillation. Equation (5) is minimized with Adam (learning rate 10^{-2}) for $6{,}000$ steps on $2{,}048$ calibration tokens drawn from the training stream, never from the evaluation data, with the encoder and decoder frozen; when the objective has not improved for 300 steps the optimizer is restarted from the best iterate with the learning rate multiplied by 0.3. For ReLU teachers the objective is zero at the identity initialization and the step is skipped. After Step 1 the RSAE is within 1–3$\%$ of the teacher's reconstruction error for all gated teachers considered.

Step 2 optimization. Training data reproduce the teachers' own pipeline (the `dictionary_learning` activation buffer): the Pile streamed in order, contexts of $1{,}024$ tokens with the first token dropped, a shuffle buffer of 244 contexts refreshed when half consumed, batches of $2{,}048$ activations, and dataset normalization $\boldsymbol{x} \leftarrow \boldsymbol{x}/\sqrt{\mathbb{E}\|\boldsymbol{x}\|_2^2}$ estimated on the first 100 batches; the first 128 documents are held out. We run $10{,}000$ Adam steps (20.5M tokens) with learning rate 3×10^{-4} (rational coefficients 3×10^{-3} for gated teachers, whose knee coefficients are $O(10^2$–$10^3)$), linear warm-up over 500 steps and linear decay from step $8{,}000$, gradient-norm clipping at 1, and, for ReLU teachers, projection of the decoder gradient onto the tangent space of the unit-norm constraint followed by row re-normalization after each step. The reconstruction term is the per-token sum of squared errors in normalized units. An auxiliary dead-feature loss in the style of AuxK (Gao et al., 2025) is added during training only: features that have not fired for 100 steps reconstruct the current residual through a linear read-out of their $d_{\text{sae}}/32$ largest pre-activations, weighted by $1/32$. It leaves the model unchanged and keeps the fraction of alive features at or above the teacher's in all settings except ReLU teachers at $d_{\text{sae}} = 2^{16}$, where it stays 2–33 points below (Table 5); without it, wide dense settings ($d_{\text{sae}} = 2^{16}$, $\ell_0 \geq 640$) collapse to 8–37$\%$ alive features. For the densest JumpReLU teachers, whose released models have fewer than 26% alive features, the auxiliary loss is disabled and training is extended to $20{,}000$ steps.

Table 4: **Sparse-probing sanity check** on Pythia-160m ReLU teacher with $\ell_0 \approx 160$ at $d_{\text{sae}} = 2^{14}$ and the RSAE fine-tuned from it, on the full SAEBench panel of 8 binary-classification tasks. We report full-dictionary probe accuracy per dataset (all SAE features, no k-sparsity constraint); higher is better.

Dataset	Teacher full ↑	RSAE full ↑	Δ
bias_in_bios_set1	95.28	95.30	+0.02
bias_in_bios_set2	93.12	93.04	−0.08
bias_in_bios_set3	90.94	91.22	+0.28
amazon_reviews	87.98	88.00	+0.02
amazon_sentiment	91.40	91.40	0.00
github-code	96.64	96.50	−0.14
ag_news	94.18	94.25	+0.07
europarl	99.96	99.96	0.00
mean	93.69	93.71	+0.02

B.3 Sparse Probing

As a sanity check that the reconstruction gains do not degrade the learned features, Table 4 reports full-dictionary linear-probe accuracy on the eight-task SAEBench panel for the Pythia-160m ReLU teacher with $\ell_0 \approx 160$ at $d_{\text{sae}} = 2^{14}$ and the RSAE fine-tuned from it. The two are on par: every per-task difference lies within ± 0.3 percentage points, comparable to the seed-to-seed noise of the probe, and the panel means are 93.71% (RSAE) and 93.69% (teacher). We make no claim that the RSAE improves interpretability; the check only indicates that its reconstruction gains are not obtained at the expense of probe accuracy.

B.4 Detailed Empirical Results

Table 5: **SAE Performance on Pythia-160m-deduped (layer 8) across three dictionary widths ($d_{\text{sae}} = \{2^{12}, 2^{14}, 2^{16}\}$), grouped by sparsity level ($\ell_0 = \{20, 40, 80, 160, 320, 640\}$).** Columns: ℓ_0, the mean number of non-zero latents per token; MSE, the mean per-token squared reconstruction error $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2$ **multiplied by 100**; alive, the percentage of latents that fire at least once on the evaluation set; ΔCE, the increase in next-token cross-entropy when the reconstruction replaces the residual stream; LR, the loss-recovered percentage relative to zero ablation.**Bold** marks the best value among the architectures within each (width, level) group.

		4k (2^{12})					16k (2^{14})					65k (2^{16})				
ℓ_0 level	SAE	ℓ_0	MSE↓	alive↑	ΔCE↓	LR↑	ℓ_0	MSE↓	alive↑	ΔCE↓	LR↑	ℓ_0	MSE↓	alive↑	ΔCE↓	LR↑
∼20	JumpReLU	19	3.34	62.2	0.579	92.2	19	2.95	40.9	0.450	93.9	19	2.69	22.0	0.378	94.9
	TopK	20	3.11	99.8	0.521	93.0	20	2.63	98.6	0.353	95.2	20	2.35	90.3	0.276	96.3
	RSAE (JumpReLU-init)	19	3.34	76.3	0.580	92.2	19	2.93	49.2	0.442	94.0	19	2.67	41.3	0.375	95.0
	RSAE (TopK-init)	20	**3.09**	**100.0**	**0.515**	**93.1**	20	**2.59**	**99.8**	**0.347**	**95.8**	20	**2.31**	**97.1**	**0.270**	**96.6**
∼40	ReLU	44	3.68	74.8	0.728	90.2	55	3.07	73.1	0.488	93.4	59	2.68	66.9	0.341	95.4
	JumpReLU	38	2.62	75.9	0.365	95.1	38	2.30	52.2	0.280	96.2	36	2.15	27.0	0.241	96.8
	TopK	40	2.50	99.9	0.339	95.4	40	2.08	99.1	0.235	96.8	40	1.82	87.6	**0.170**	97.7
	RSAE (ReLU-init)	44	2.75	98.3	0.421	94.3	53	2.26	92.4	0.323	95.6	59	1.87	65.0	0.193	97.4
	RSAE (JumpReLU-init)	38	2.62	96.9	0.364	95.1	38	2.28	90.5	0.277	96.3	36	2.14	62.1	0.239	96.8
	RSAE (TopK-init)	40	**2.47**	**100.0**	**0.327**	**95.6**	40	**2.03**	**99.9**	**0.234**	**97.0**	40	**1.78**	**100.0**	0.171	**97.9**
∼80	ReLU	79	3.04	75.9	0.526	92.9	102	2.51	74.0	0.341	95.4	115	2.18	68.7	0.246	96.7
	JumpReLU	75	1.98	87.9	0.221	97.0	68	1.85	51.3	0.195	97.4	70	1.71	24.2	0.168	97.7
	TopK	80	1.91	**100.0**	0.222	97.0	80	1.59	99.3	0.154	97.9	80	1.41	79.6	0.114	**98.5**
	RSAE (ReLU-init)	79	2.12	98.7	0.276	96.3	102	1.82	98.4	0.220	97.0	115	1.42	65.6	0.117	98.4
	RSAE (JumpReLU-init)	75	1.96	99.9	0.219	97.0	68	1.81	90.7	0.178	97.6	70	1.63	37.2	0.153	98.1
	RSAE (TopK-init)	79	**1.88**	**100.0**	**0.206**	**97.2**	79	**1.54**	**100.0**	**0.134**	**98.2**	80	**1.36**	**100.0**	**0.109**	**98.5**
∼160	ReLU	122	2.60	76.2	0.404	94.5	164	2.12	74.4	0.258	96.5	192	1.84	69.3	0.183	97.5
	JumpReLU	155	1.34	81.0	0.116	98.4	150	1.26	41.5	0.107	98.6	125	1.35	17.5	0.120	98.4
	TopK	160	1.30	99.5	0.125	98.3	160	1.10	94.6	0.093	**98.8**	160	0.99	66.1	0.071	**99.0**
	RSAE (ReLU-init)	121	1.69	99.9	0.187	97.5	160	1.38	98.9	0.141	98.1	185	1.08	64.5	0.072	**99.0**
	RSAE (JumpReLU-init)	154	1.31	**100.0**	0.106	98.5	150	1.22	95.6	0.097	98.6	125	1.32	45.0	0.117	98.4
	RSAE (TopK-init)	160	**1.24**	99.9	**0.105**	**98.6**	161	**1.05**	**100.0**	**0.092**	**98.8**	160	**0.96**	**99.8**	**0.069**	**99.0**
∼320	ReLU	224	1.98	76.3	0.259	96.5	319	1.60	74.7	0.165	97.8	410	1.37	70.0	0.111	98.5
	JumpReLU	326	0.56	63.7	0.033	99.5	314	0.60	26.1	0.036	99.5	291	0.69	8.7	0.044	99.4
	TopK	320	0.54	66.3	0.038	99.5	320	0.52	46.3	0.036	99.5	320	0.51	25.5	0.034	99.5
	RSAE (ReLU-init)	224	1.08	99.2	0.097	98.7	319	0.80	99.2	0.064	99.1	389	0.57	60.1	**0.029**	**99.6**
	RSAE (JumpReLU-init)	326	0.54	98.8	0.033	**99.6**	314	0.58	56.6	0.035	99.5	291	0.67	30.5	0.043	99.4
	RSAE (TopK-init)	320	**0.50**	**99.3**	**0.032**	**99.6**	320	**0.50**	**99.9**	**0.033**	**99.6**	320	**0.49**	**98.7**	0.033	99.4
∼640	ReLU	336	1.55	76.4	0.176	97.6	501	1.26	74.9	0.114	98.5	682	1.07	70.4	0.076	99.0
	JumpReLU	626	**0.12**	25.6	**0.007**	**99.9**	607	0.13	7.0	0.008	99.9	614	0.26	1.7	0.010	99.9
	TopK	640	0.15	23.3	0.010	**99.9**	639	0.15	6.8	0.009	99.9	640	0.16	2.6	0.009	99.9
	RSAE (ReLU-init)	336	0.58	96.6	0.045	99.4	501	0.31	83.0	0.018	99.8	648	**0.14**	37.1	**0.005**	**99.9**
	RSAE (JumpReLU-init)	623	**0.12**	25.4	**0.007**	**99.9**	607	0.12	63.0	0.008	99.9	614	0.25	20.1	0.010	**99.9**
	RSAE (TopK-init)	639	0.13	**100.0**	**0.007**	**99.9**	642	0.13	**100.0**	0.009	99.9	640	0.15	**95.3**	0.009	**99.9**