CLASSAE: CLASS-ALIGNED SPARSE AUTOENCODERS VIA DIFFERENTIABLE FEATURE-CLASS AFFINITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse Autoencoders (SAEs) began as an unsupervised tool for decomposing neural representations into sparse, interpretable features, and are increasingly used not only for passive analysis but also for active interventions such as unlearning, bias mitigation, and concept editing. A central challenge for these editing and steering methods is reliably matching features to target concepts; most current approaches address this by computing post-hoc scores over an already-trained, frozen dictionary. We instead introduce ClasSAE, a novel method that both automatically assigns classes to features and guides the encoder toward class-separable representations during training. Specifically, we apply a differentiable top-k operator to a trainable feature–class affinity matrix with per-feature budgets, coupling the features selected for each sample to the classes they are trained to represent. Because gradients flow through the selection of active features rather than only through their magnitudes, the encoder and the affinity matrix co-adapt rather than being fit in separate stages. The result is a dictionary that is both class-separable and class-annotated, with no need for post-hoc probing. We propose three variants for enforcing sparsity within this framework, which achieve comparable overall performance with slightly different trade-offs. Using CLIP ViT-L/14 embeddings on ImageNet, we show that the learned affinity matrix agrees closely with an independently estimated post-hoc feature-class matrix computed on held-out data. The model also supports direct class prediction from the encoder and affinity matrix alone, without a separately fitted classifier, and its more class-aligned encoder yields improved separation in Targeted Probe Perturbation evaluations. These results show that known concept annotations can be used not merely to locate relevant features in an SAE dictionary after training, but also to shape the dictionary itself during learning.

1 INTRODUCTION

Sparse autoencoders (SAEs) have become a leading tool for mechanistic interpretability, decomposing dense activations into sparse, putatively monosemantic features in a purely unsupervised way (Huben et al., 2024; Bricken et al., 2023; Templeton et al., 2024). However, recent work questions whether this unsupervised decomposition is even well-defined: Leask et al. (2025) show that SAEs do not converge to a canonical set of ground-truth units, and different random seeds or hyperparameters yield dictionaries that disagree substantially with one another (Paulo & Belrose, 2026).

Despite this, SAEs are widely used not only for analysis but also for *intervention*, including activation steering, unlearning, and bias mitigation (Cywiński & Deja, 2025; Ahsan & Wallace, 2026; Cho & Hockenmaier, 2025). All of these methods work by editing the small subset of features responsible for a target concept. This is exactly where the unsupervised nature of SAEs becomes a liability: since training has no access to the concepts practitioners care about, every such application must first solve a separate post-hoc matching problem, probing a frozen dictionary to infer which features correspond to which concept. This two-stage recipe inherits the instability noted above: a target concept may end up split across several features, entangled with other concepts, or absent from the dictionary altogether, and nothing during training prevents this.

We observe that in most of these intervention settings, the relevant concepts are not in fact unknown: unlearning a category or mitigating bias along a known attribute both presuppose that the practitioner

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.

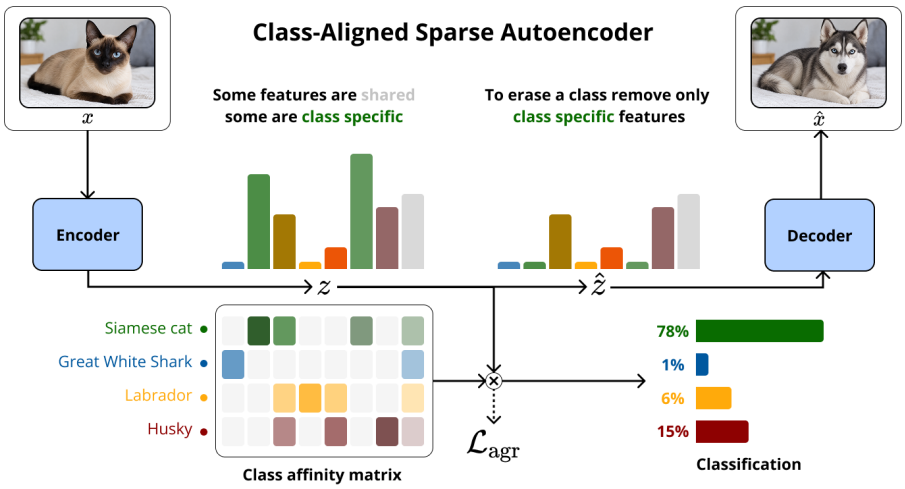

Figure 1: ClasSAE jointly learns a sparse dictionary and a class affinity matrix, enabling classifier-free class prediction and targeted editing: removing only the features exclusive to a class erases that class from the reconstruction while shared features are preserved.

knows in advance which classes should be disentangled. Rather than hoping that an unsupervised SAE will isolate these concepts and searching for them afterwards, we argue that this knowledge should directly shape training.

To this end, we introduce **ClasSAE**, an SAE that learns a class-annotated, class-separable dictionary directly during training rather than relying on post-hoc analysis. We provide a general visual overview of the proposed framework in Figure 1. Our contributions are as follows:

- **Automatic feature–class association.** We learn a feature–class affinity matrix jointly with the SAE, removing the need for post-hoc probing to identify which features correspond to which class.
- **Class-aligned encoder–decoder.** The encoder and decoder are trained so that the dictionary is class-separable by construction, rather than merely reconstruction-optimal with class structure emerging only incidentally.
- **Three sparsity variants under a shared framework.** We propose three ways of enforcing sparsity in the latent representation, which trade off reconstruction quality against class alignment.
- **A single joint training objective.** We combine these mechanisms into one end-to-end objective, so the dictionary and the affinity matrix co-adapt within one training process rather than being fit in separate stages. This general framework yields a single model with multiple applications: classification (Section 4), interpretability (Figure 2), and targeted concept editing (Section 4).
- **Experimental evaluation.** We assess the agreement of ClasSAE's affinity matrix with an independently estimated post-hoc feature–class matrix, its ability to support classifier-free class prediction, and its performance on Targeted Probe Perturbation, comparing against other SAE architectures and variants of our own method.

2 Related Work

Sparse autoencoders and their failure modes. SAEs decompose activations into sparse linear combinations of learned directions (Sharkey et al., 2022; Bricken et al., 2023; Huben et al., 2024) and scale to frontier models (Templeton et al., 2024). Architectural work has concentrated on the sparsity constraint: k-sparse and top-k SAEs enforce an exact count (Makhzani & Frey, 2014; Gao et al., 2024); Gated SAEs decouple *which* features fire from *how strongly* (Rajamanoharan et al., 2024a); JumpReLU uses a discontinuous threshold (Rajamanoharan et al., 2024b); BatchTopK moves

the budget from the sample to the batch, so individual samples may use more or fewer latents at fixed average sparsity (Bussmann et al., 2024); Matryoshka SAEs impose a nested hierarchy over dictionary prefixes (Bussmann et al., 2025; Zaigrajew et al., 2025). Lee et al. (2025) instead enforce approximate quasi-orthogonality among dictionary elements, which enables a Top-AFA activation function that removes the k hyperparameter entirely by choosing the number of active features whose reconstruction most closely matches the input's L^2 norm.

A parallel line documents what SAEs get wrong: SAEs do not reliably decompose concepts into canonical units (Leask et al., 2025), and apparently monosemantic latents are *absorbed* into their children (Chanin et al., 2025); dictionaries are unstable across seeds, motivating constrained parameterizations (Fel et al., 2025); much of the residual is linearly predictable "dark matter" (Engels et al., 2025); and the i.i.d. prior of per-token SAEs conflicts with the temporal structure of language (Bhalla et al., 2026; Lubana et al., 2026). SAEBench (Karvonen et al., 2025) consolidates several of these diagnostics, one of which we adopt in our experiments (see Section 4). We hypothesize that these failure modes stem, at least in part, from the absence of any external signal guiding the dictionary toward useful structure; the class-alignment mechanism we introduce in this work is one candidate for such a signal, though our focus here is on evaluating its practical, measurable benefits rather than directly testing this broader hypothesis.

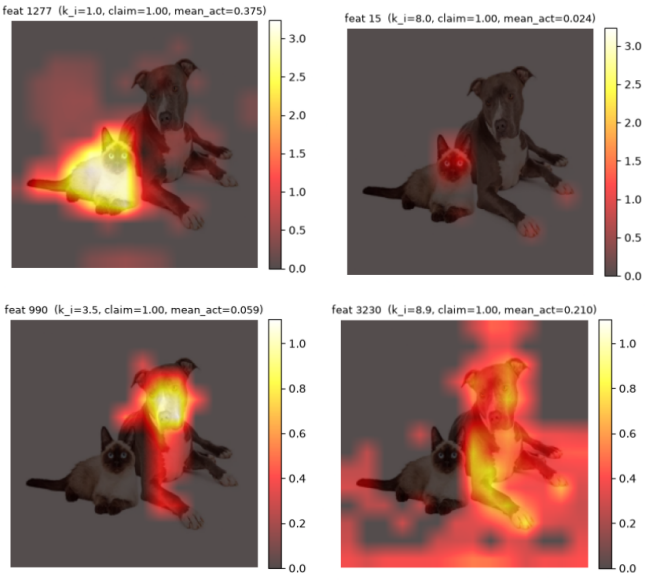

Figure 2: Feature activation heatmaps for ClasSAE trained on DINOv3 patch tokens, using whole-image class labels only. **Top:** features claimed by "Siamese cat, Siamese." **Bottom:** features claimed by "American Staffordshire terrier, Staffordshire terrier, American pit bull terrier, pit bull terrier." Each panel reports the feature's budget k_i, its claim strength for the labeled class, and its mean activation. Details in Section F.

Automatic feature interpretation and labeling. A separate line automates the post-hoc labeling step itself. Bricken et al. (2023) establish the basic dictionary-learning setup that makes such labeling meaningful in the first place. Paulo et al. (2025) generate natural-language feature explanations at scale using LLMs and, notably, evaluate them via intervention rather than input examples alone. Gur-Arieh et al. (2025) show that input-only explanations are incomplete, since a feature's meaning also depends on its effect on the output, and propose output-centric descriptions from steering/unembedding, finding input- and output-centric signals complementary. SALVE (Flovik, 2026) similarly correlates SAE features with visual concepts, using Grad-CAM (Selvaraju et al., 2016) and activation maximization to explain individual features. All of these approaches treat labeling as a separate step applied to an already-trained, frozen dictionary; ClasSAE instead produces class annotations as a direct byproduct of training, with no explanation pipeline required.

SAEs as intervention mechanisms. SAEs are increasingly used to directly edit model behavior. SAEuron (Cywiński & Deja, 2025) trains a BatchTopK SAE and scores features to find those tied to a target concept, then ablates or negatively scales them to unlearn it from diffusion models. SAEmnesia (Cassano et al., 2026b) is closest in spirit to ClasSAE: a two-stage scheme first trains an unsupervised SAE, then collapses each class onto a single latent coordinate. ClasSAE differs on both counts, using one joint objective rather than two stages, and a per-feature affinity matrix rather than one coordinate per class, so features can be shared across classes rather than duplicated. This avoids wasting capacity and, for unlearning, lets us ablate only class-exclusive features while shared features are preserved for retain accuracy. Cassano et al. (2026a) find SAEs good at locating concepts in diffusion models but show that direct feature ablation pushes activations off-manifold, causing artifacts, and use the located

features to guide patch-swapping instead. CRISP (Ashuach et al., 2026) trains a LoRA adapter (Hu et al., 2022) guided by SAE features to suppress a target concept while preserving unrelated ones in LLM unlearning. Ahsan & Wallace (2026) apply the locate-then-ablate recipe to detect and mitigate racial bias in healthcare LLMs via probing. Unlike these approaches, which first train an SAE and then annotate its features, ClasSAE folds both goals into a single training objective.

Concept supervision and differentiable selection. Relaxations of sorting and top-k include Gumbel reparameterizations (Jang et al., 2017) and optimal-transport formulations (Xie et al., 2020); we use LapSUM (Struski et al., 2025), which accepts a budget and returns weights summing exactly to it, giving a differentiable selection signal for the per-feature k_i in ClasSAE. Concept activation vectors and interpretable basis decompositions map latent spaces to human-named ideas (Kim et al., 2018; Zhou et al., 2018), later automated (Ghorbani et al., 2019; Bykov et al., 2023), while Concept Bottleneck and Concept Embedding Models enforce such representations during training (Koh et al., 2020; Espinosa Zarlenga et al., 2022). ClasSAE sits between these traditions: the encoder and decoder stay unsupervised at inference, and only a budgeted auxiliary head carries class semantics.

3 METHOD

Preliminaries Let $x \in \mathbb{R}^n$ be an activation extracted from a pretrained network and $d \gg n$ the dictionary size. With a pre-encoder bias $b_{\text{pre}} \in \mathbb{R}^n$, encoder weights $W_{\text{enc}} \in \mathbb{R}^{d\times n}$, encoder bias $b_{\text{enc}} \in \mathbb{R}^d$ and decoder $W_{\text{dec}} \in \mathbb{R}^{n\times d}$,

$$z = \mathrm{ReLU}(W_{\text{enc}}(x - b_{\text{pre}}) + b_{\text{enc}}) \in \mathbb{R}^d_{\geq 0}, \qquad \hat{x} = W_{\text{dec}}\, f(z) + b_{\text{pre}}, \tag{1}$$

where f is the sparsification and the columns of W_{dec} form the dictionary. Training minimizes $\|x-\hat{x}\|_2^2$ under a sparsity mechanism: an L^1 penalty, a threshold, or a hard top-k that fixes $\|f(z)\|_0 = k$ for every input.

Soft top-k We use the differentiable operator of Struski et al. (2025), which we state explicitly since the formal definition matters below. Let ψ be a smooth, strictly increasing CDF (the Laplace CDF in Struski et al., 2025) and $\alpha > 0$ a temperature. For $v \in \mathbb{R}^m$ and a budget $\beta \in (0, m)$ there is a unique threshold $t = t(v, \beta, \alpha)$ solving $\sum_j \psi((v_j - t)/\alpha) = \beta$, and

$$\mathrm{SoftTopK}(v, \beta, \alpha)_j \;=\; \psi\!\left(\frac{v_j - t}{\alpha}\right) \in (0, 1), \qquad \sum_{j=1}^{m} \mathrm{SoftTopK}(v, \beta, \alpha)_j = \beta. \tag{2}$$

Two properties are used throughout: the budget β need not be an integer, and as $\alpha \to 0$ the output concentrates on the $\lceil\beta\rceil$ largest entries. Both t and the outputs are differentiable in v, β and α.

A class affinity matrix: per-feature annotation We begin by constructing an automatic feature annotation mechanism. Let C be the number of classes and $\Lambda \in \mathbb{R}^{d\times C}$ a trainable matrix of *class logits*: Λ_{ic} says how much coordinate i "wants" class c. Raw logits mean nothing on their own; what gives them meaning is that claims are scarce. We fix a global association budget $K_{\text{tot}} = \rho\, d$ (on average ρ classes per coordinate) and let a trainable vector $\kappa \in \mathbb{R}^d$ split it:

$$k \;=\; K_{\text{tot}} \cdot \mathrm{softmax}(\kappa) \in \mathbb{R}^d_{>0}, \qquad \textstyle\sum_i k_i = K_{\text{tot}}, \qquad M_i = \mathrm{SoftTopK}(\Lambda_i, k_i, \alpha_M) \in (0,1)^C. \tag{3}$$

Row-wise, M_i answers "which classes does this feature serve"; column-wise, "which features serve this class". Nothing forces the k_i to be equal.[1] Since M depends only on parameters, Equation (3) is one batched operator call per step on a $d \times C$ matrix, i.e. 16.4M entries at $d = 16384$, $C = 1000$, which is negligible beside the encoder.

Three sparsity variants The sparsity mechanism is a central design choice in any SAE. We propose three alternatives, each achieving comparable performance with different trade-offs, discussed in Section 4. Variants (i) and (ii) share the same forward-pass budget (Bussmann et al., 2024), zeroing

[1] We clip $k_i \leq C$ for numerical safety, since SoftTopK's budget cannot exceed the dimension it selects over; in practice a well-trained model never approaches this bound, as doing so would waste budget with no benefit to any row's agreement score.

every coordinate outside the top $\bar{k} \cdot B$ activations in a batch of size B, where $\bar{k} \ll d$ is the target L^0, and differ only in how gradients reach the encoder; variant (iii) uses no thresholding operator at all.

(i) Soft-gradient BatchTopK. We replace the identity straight-through estimator with a gradient borrowed from a soft surrogate,

$$f'(z) = p(x) \odot z, \qquad p(x) = \mathrm{SoftTopK}\left(z, \hat{k}(x), \alpha_p\right) \in (0,1)^d, \qquad \hat{k}(x) = \|f(z)\|_0, \tag{4}$$

and use $f(z) - \mathrm{stop_grad}\big(f'(z)\big) + f'(z)$ as the representation: the forward pass is exactly hard BatchTopK, while the backward pass flows through the soft ranking, giving gradient signal through *selection* rather than only through the surviving coordinates' scale.

(ii) Hard BatchTopK. The standard estimator: $f(z)$ forward, identity gradient on active coordinates, zero elsewhere.

(iii) L^1-relaxed. No thresholding operator at all; sparsity is encouraged only via an L^1 penalty added to the loss, with the encoder otherwise dense.

Remark 1. Variant (i) may look like it pays for an extra SoftTopK call that (ii) and (iii) avoid. It does not: every variant must compute $p(x)$ regardless [2] , since the agreement loss explained in the next paragraph requires it.

Coupling the two budgets Both $p(x)$ and M are budgeted selections, so we ask them to agree. We use the *selection mass*, not the activation magnitude, normalizing p by its own budget:

$$\pi(x) = \frac{p(x)}{\hat{k}(x)} \in \Delta^{d-1}, \qquad s_c(x) = \big(M^\top \pi(x)\big)_c = \frac{1}{\hat{k}} \sum_i p_i M_{ic} \in [0,1]. \tag{5}$$

By the defining property of Equation (2), $\pi(x)$ lands on the simplex exactly, with no ε and no dependence on the scale of the activations. The agreement loss over a batch $\{(x^{(b)}, c_b)\}_{b \le B}$ is

$$\mathcal{L}_{\mathrm{agr}} = -\frac{1}{B} \sum_{b=1}^{B} s_{c_b}^{(b)}, \qquad \mathcal{L}(x) = \|x - \hat{x}\|_2^2 + \gamma\, \mathcal{L}_{\mathrm{aux}} + \mu\, \mathcal{L}_{\mathrm{agr}}. \tag{6}$$

where $\mathcal{L}_{\mathrm{aux}}$ is a standard dead-latent loss proposed by Gao et al. (2024). For variant (iii) we also add L^1 penalty as mentioned earlier. Gradients flow both ways: forward into the annotation (Λ: which classes, κ: how many) and backward through p into the encoder. Without the second path, this would be post-hoc labeling with extra steps. Figure 3 visualizes this mechanism.

Remark 2 (Why p and not z). The construction is about which features a sample uses, not how strongly they fire. Magnitudes are set by reconstruction and decoder-column norms, so a score built on z lets one large activation dominate a sample that selected $\hat{k} = 40$ features. With p, each selected feature contributes ≈ 1 and s_c is exactly the claimed fraction of the selection; gradients then reach the encoder through the ranking that $\mathrm{SoftTopK}$ differentiates, pushing class-consistent features *up the order* rather than competing with reconstruction over magnitudes.

Remark 3 (Precision, not recall). In the hard limit both objects are sets and $s_c = |S(x) \cap F_c|/|S(x)|$ is a *precision*, maximal as soon as $S(x) \subseteq F_c$: a class may own 40 features while any sample uses 8. The soft version inherits this, since $\sum_i p_i = \hat{k}$ gives $1 - s_c = \hat{k}^{-1} \sum_i p_i (1 - M_{ic})$, a condition on the selected coordinates only. Normalizing by $|F_c|$ instead would make s_c a recall, force every sample to fire everything its class owns, reward larger $\hat{k}$, and suppress the adaptivity we want to preserve.

What the annotation head actually optimizes Write the class-balanced form of Equation (6) using $\bar{A}_{ic} = |B_c|^{-1} \sum_{b \in B_c} \pi_i^{(b)}$, the average share of class c's selection budget spent on coordinate i. Then $\mathcal{L}_{\mathrm{agr}} = -\frac{1}{C} \langle M, \bar{A} \rangle$, and since each $\pi^{(b)}$ lies on the simplex, $\bar{A}$ is column-stochastic, so $\mathcal{L}_{\mathrm{agr}} \in [-1, 0]$ with -1 meaning perfect agreement.

Proposition 1 (Water-filling). *Because $k_i = \sum_c M_{ic}$, the budget variable can be eliminated: the set of tables reachable by Equation* (3) *in the limit is exactly $\mathcal{M} = \{M \in [0,1]^{d \times C} : \sum_{i,c} M_{ic} = K_{\mathrm{tot}}\}$,*

[2] Note that $p(x)$ can be computed for any SAE's z; we use this fact in our experiments involving post-hoc estimated matrices in Section 4.

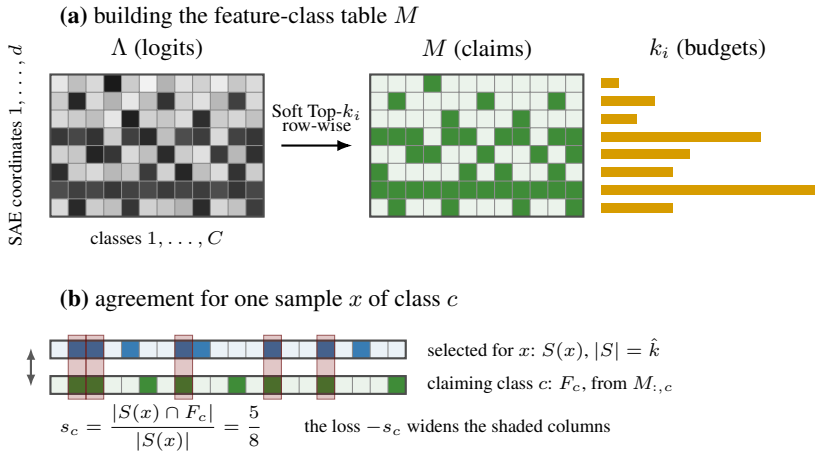

Figure 3: **(a)** Each dictionary coordinate holds a row of trainable class logits Λ_i (grey). A trainable vector κ splits a *global* association budget $K_{\text{tot}} = \rho d$ into per-feature budgets k_i (orange), and Soft Top-k_i turns each row into a claim M_i carrying exactly k_i units of mass (green). Budgets need not be equal: row 1 is a specialist ($k = 1$), row 7 a generalist ($k = 12$), and what one row takes another must give up. **(b)** For a sample of class c, the SAE selects $\hat{k}$ coordinates (blue) while column $M_{:,c}$ marks the coordinates that claimed c (green). The score is the fraction of the selected set that is claimed (shaded); the double arrow indicates that minimizing $-s_c$ pushes from both sides at once, since the encoder must rank claimed features into the top-$\hat{k}$ while the table must claim features that actually fire. Both panels are drawn in the hard limit; during training $S(x)$ and F_c are the soft weights $\pi(x)$ and $M_{:,c}$.

on which $\langle M, \bar{A} \rangle$ is linear. Hence for a fixed encoder the optimum is attained at a vertex: $M^\star$ is the indicator of the K_{tot} largest entries of $\bar{A}$, and $k_i^\star = |\{c : \bar{A}_{ic} > \nu\}|$ for a single global threshold ν set by K_{tot}.

A proof is in Section A. Three consequences follow.

(i) The intended behavior is the optimum. A coordinate claims as many classes as actually spend budget on it, so specialists receive $k_i \approx 1$ and shared features $k_i \gg \rho$ with no extra mechanism.

(ii) The per-row budget is a reparameterization, not a constraint. A flat variant applying one call to $\text{vec}\,\Lambda$ with budget K_{tot} has the same feasible set and optimum, so splitting by row costs nothing in principle. We do not ablate this directly: a single SoftTopK call over the flattened $d \times C$ vector is numerically impractical at our scale, so the per-row split is an engineering solution that recovers the same optimum through cheaper, parallel, per-row calls, not a capacity trade-off we verify empirically.

(iii) Rare features would lose their annotation at the idealized vertex. Uniformly small rows of $\bar{A}$ fall below ν and get $k_i \to 0$ at this solution, mirroring the dead-latent pathology $\mathcal{L}_{\text{aux}}$ guards against on the activation side. We are aware of this prediction but do not observe it in practice: annotated rare features remain qualitatively sharp rather than starved. Our hypothesis, not a claim we verify, is that this is because Proposition 1 describes a fixed-encoder optimum against a static $\bar{A}$, while in joint training $\bar{A}$ moves with the encoder and $k_i = K_{\text{tot}} \cdot \text{softmax}(\kappa)_i$ is a smooth reparameterization with no pressure to reach its limiting vertex against a non-stationary target; we remain unsure whether this is the full explanation.

Contrastive form Equation (6) rewards the true class but never penalizes the others, so only the budget stops M from drifting toward a uniform matrix. Normalizing over classes repairs this: $\mathcal{L}_{\text{agr}}^{\text{CE}} = \frac{1}{B} \sum_b [-s_{c_b}^{(b)}/\tau + \log \sum_{c'} \exp(s_{c'}^{(b)}/\tau)]$, which equals $\mathcal{L}_{\text{agr}}/\tau$ plus a log-sum-exp term and, as $\tau \to \infty$, recovers Equation (6) with the class mean subtracted, up to scaling by τ and an additive constant $\log C$. Under it M becomes a budget-constrained sparse linear classifier on the SAE code, giving a free evaluation metric (Section 4). It is not linear in M, so Proposition 1 describes only the base objective; we observe the same specialist/generalist behavior and the absence of annotation collapse under this form as well. Given these properties, we adopt $\mathcal{L}_{\text{agr}}^{\text{CE}}$ as the default agreement

Table 1: **Reconstruction quality.** Comparison at matched target $L_0 \approx 380$. L_0 is the realized L^0-norm of the code; **FVE** is the fraction of input variance explained by the reconstruction; **Alive** is the fraction of dictionary coordinates that fire on at least one validation sample. Bold marks the best result among architectures matched to the target L_0; †Top-AFA selects its own L_0 and is excluded from this comparison.

Architecture	L_0	FVE ↑	Alive ↑
BatchTopK	$381.055_{\pm 0.073}$	$0.973_{\pm 0.000}$	$0.713_{\pm 0.013}$
JumpReLU	$383.346_{\pm 3.158}$	$\mathbf{0.986}_{\pm 0.000}$	$0.579_{\pm 0.019}$
Matryoshka	$381.001_{\pm 0.199}$	$0.968_{\pm 0.000}$	$0.377_{\pm 0.027}$
Standard	$345.936_{\pm 1.021}$	$0.899_{\pm 0.000}$	$0.671_{\pm 0.005}$
Top-AFA†	$1184.456_{\pm 24.535}$	$0.991_{\pm 0.001}$	$0.662_{\pm 0.029}$
TopK	$380.000_{\pm 0.000}$	$0.970_{\pm 0.000}$	$0.455_{\pm 0.010}$
ClasSAE Shuffle	$380.715_{\pm 0.095}$	$0.977_{\pm 0.000}$	$0.522_{\pm 0.021}$
ClasSAE	$379.502_{\pm 0.133}$	$0.934_{\pm 0.000}$	$\mathbf{1.000}_{\pm 0.000}$
ClasSAE Hard	$377.967_{\pm 0.238}$	$0.910_{\pm 0.000}$	$\mathbf{1.000}_{\pm 0.000}$
ClasSAE L^1	$444.028_{\pm 0.408}$	$0.864_{\pm 0.000}$	$\mathbf{1.000}_{\pm 0.000}$

loss in all experiments unless stated otherwise (we give the derivation and BatchTopK correction in Section B).

Proposition 2 (PMI equivalence)**.** *In the hard limit $\alpha_p \to 0$ and with support size held constant across samples ($\hat{k}(x) \equiv \bar{k}$), the water-filling optimum of Proposition 1 converges to a per-feature PMI threshold: writing $\mathrm{PMI}(i,c) = \log \frac{P(A_i=1|c)}{P(A_i=1)}$ for the hard support indicator $A_i(x) = \mathbf{1}[i \in S(x)]$,*

$$M^\star_{ic} \;\longrightarrow\; \mathbf{1}\big[\mathrm{PMI}(i,c) > \nu_i\big], \qquad \nu_i := \log(\nu\bar{k}) - \log P(A_i{=}1) \tag{7}$$

for the same global threshold ν as Proposition 1. That is, the optimal annotation matrix is provably a pointwise-mutual-information thresholding rule, with a per-feature calibrated cutoff ν_i that falls as feature i's marginal firing rate grows.

4 Experiments

We evaluate ClasSAE on four questions: does the agreement loss cost reconstruction quality, is the learned annotation M faithful to held-out data, can the SAE and M serve directly as a classifier, and does the encoder itself become better structured for downstream editing?

Setup All experiments use CLIP ViT-L/14 embeddings of ImageNet-1k (Deng et al., 2009), training on the train split and evaluating on the validation split. We use dictionary size $d = 16384$ and association budget $\rho = 40$ throughout; remaining hyperparameters are given in Section C. We denote the three sparsity variants of Section 3 as ClasSAE (soft-gradient BatchTopK), ClasSAE Hard, and ClasSAE L^1. ClasSAE Shuffle is a control that trains the soft-gradient variant with randomly permuted class labels. We train every architecture with 5 seeds and report the mean and standard deviation of each metric across seeds.

Reconstruction Quality This is a standard reconstruction/sparsity comparison, included to establish that ClasSAE sits in a reasonable operating range relative to existing SAE architectures. We report fraction of variance explained $\mathrm{FVE} = 1 - \frac{\sum_j \mathrm{Var}(x_j - \hat{x}_j)}{\sum_j \mathrm{Var}(x_j)}$, alongside the realized L_0 and the fraction of alive latents in Table 1. Most architectures are matched to the target $L_0 \approx 380$ directly by their sparsity mechanism; Standard and ClasSAE L^1 instead reach this target indirectly, via an L^1 coefficient tuned to land near it, which accounts for their larger deviation. Top-AFA is deliberately left unmatched: its premise is choosing its own per-input L_0 from the input's L^2 norm, so constraining it to $\bar{k} = 380$ would defeat the mechanism being evaluated. The agreement loss does lower FVE relative to the unsupervised baselines (0.934 for ClasSAE versus 0.973 for BatchTopK, which shares its forward pass), while ClasSAE Shuffle (0.977) shows that this cost comes from the class signal rather than from the mechanism itself. Only the ClasSAE variants trained with true labels reach full dictionary utilization (Alive $= 1.000$), which we attribute to the affinity matrix giving every coordinate a class to serve.

Table 2: M**-honesty.** Mean/median cosine similarity (CS) and precision/recall/F1 between the train-side matrix (M for ClasSAE, post-hoc $\hat{M}$ for baselines) and the test-side $\hat{M}$. ClasSAE variants demonstrate that M carries a real, class-faithful association, not just an artifact of training.

Architecture	Mean CS ↑	Median CS ↑	Mean Precision ↑	Mean Recall ↑	Mean F1 ↑
BatchTopK	$0.410_{\pm 0.009}$	$0.269_{\pm 0.043}$	$0.348_{\pm 0.007}$	$0.347_{\pm 0.008}$	$0.346_{\pm 0.008}$
JumpReLU	$0.287_{\pm 0.010}$	$0.000_{\pm 0.000}$	$0.465_{\pm 0.016}$	$0.463_{\pm 0.016}$	$0.463_{\pm 0.016}$
Matryoshka	$0.092_{\pm 0.002}$	$0.000_{\pm 0.000}$	$0.178_{\pm 0.018}$	$0.178_{\pm 0.017}$	$0.178_{\pm 0.017}$
Standard	$0.555_{\pm 0.004}$	$0.848_{\pm 0.000}$	$0.683_{\pm 0.004}$	$0.681_{\pm 0.004}$	$0.682_{\pm 0.004}$
Top-AFA	$0.266_{\pm 0.016}$	$0.127_{\pm 0.048}$	$0.180_{\pm 0.005}$	$0.179_{\pm 0.005}$	$0.179_{\pm 0.005}$
TopK	$0.195_{\pm 0.003}$	$0.000_{\pm 0.000}$	$0.177_{\pm 0.004}$	$0.176_{\pm 0.004}$	$0.176_{\pm 0.004}$
ClasSAE Shuffle	$0.017_{\pm 0.000}$	$0.000_{\pm 0.000}$	$0.034_{\pm 0.000}$	$0.042_{\pm 0.002}$	$0.009_{\pm 0.000}$
ClasSAE	$0.822_{\pm 0.001}$	$0.827_{\pm 0.001}$	$0.766_{\pm 0.002}$	$\mathbf{0.889}_{\pm 0.001}$	$0.807_{\pm 0.001}$
ClasSAE Hard	$0.847_{\pm 0.001}$	$0.850_{\pm 0.001}$	$\mathbf{0.835}_{\pm 0.001}$	$0.883_{\pm 0.001}$	$\mathbf{0.851}_{\pm 0.001}$
ClasSAE L^1	$\mathbf{0.851}_{\pm 0.001}$	$\mathbf{0.891}_{\pm 0.001}$	$0.812_{\pm 0.001}$	$0.823_{\pm 0.000}$	$0.796_{\pm 0.001}$

Table 3: **Out-of-the-box classification. Acc**, **Acc Top 5**, **Acc Top 10**: top-1, top-5, and top-10 accuracy. **F1**: macro F1. **Margin**: gap between the top two class scores. ClasSAE L^1 is the top scorer on every metric, and all ClasSAE variants trained with true labels achieve a larger margin than any baseline. Standard is the strongest baseline: it outperforms the soft-gradient ClasSAE on accuracy and F1 and slightly exceeds ClasSAE Hard on Acc Top 10.

Architecture	Acc ↑	Acc Top 5 ↑	Acc Top 10 ↑	F1 ↑	Margin ↑
BatchTopK	$0.002_{\pm 0.000}$	$0.007_{\pm 0.000}$	$0.015_{\pm 0.001}$	$0.000_{\pm 0.000}$	$-0.148_{\pm 0.001}$
JumpReLU	$0.678_{\pm 0.003}$	$0.855_{\pm 0.004}$	$0.893_{\pm 0.004}$	$0.682_{\pm 0.002}$	$0.023_{\pm 0.000}$
Matryoshka	$0.002_{\pm 0.000}$	$0.006_{\pm 0.000}$	$0.015_{\pm 0.000}$	$0.000_{\pm 0.000}$	$-0.137_{\pm 0.002}$
Standard	$0.760_{\pm 0.000}$	$0.933_{\pm 0.000}$	$0.963_{\pm 0.000}$	$0.757_{\pm 0.000}$	$0.013_{\pm 0.000}$
Top-AFA	$0.004_{\pm 0.002}$	$0.012_{\pm 0.002}$	$0.022_{\pm 0.005}$	$0.003_{\pm 0.001}$	$-0.044_{\pm 0.007}$
TopK	$0.611_{\pm 0.008}$	$0.779_{\pm 0.010}$	$0.821_{\pm 0.008}$	$0.607_{\pm 0.004}$	$0.006_{\pm 0.000}$
ClasSAE Shuffle	$0.001_{\pm 0.000}$	$0.005_{\pm 0.001}$	$0.010_{\pm 0.001}$	$0.000_{\pm 0.000}$	$-0.060_{\pm 0.002}$
ClasSAE	$0.750_{\pm 0.002}$	$0.914_{\pm 0.003}$	$0.941_{\pm 0.001}$	$0.748_{\pm 0.003}$	$0.096_{\pm 0.001}$
ClasSAE Hard	$0.789_{\pm 0.001}$	$0.940_{\pm 0.000}$	$0.962_{\pm 0.000}$	$0.787_{\pm 0.001}$	$0.131_{\pm 0.001}$
ClasSAE L^1	$\mathbf{0.799}_{\pm 0.001}$	$\mathbf{0.955}_{\pm 0.000}$	$\mathbf{0.976}_{\pm 0.000}$	$\mathbf{0.796}_{\pm 0.001}$	$\mathbf{0.168}_{\pm 0.000}$

M-Honesty Here we ask whether the trained M is a faithful description of class structure on unseen data, not just a byproduct of fitting the training set. As a post-hoc reference we compute $\hat{M}$, a PMI matrix over the same feature-class pairs, with each row's entries below its own k_i/d threshold zeroed out to give it the same row-wise budget structure as M. We compare the train-side matrix (M for ClasSAE, $\hat{M}$ for baselines) against the test-side $\hat{M}$ computed independently on the ImageNet-1k validation set.

Cosine similarity is computed directly, row-wise, between the train-side and test-side matrices. For precision, recall, and F1, each feature's *claimed* set is the top-k_i classes of its row; ground truth is this set computed on test data, prediction is the same set computed on train data.

As Table 2 shows, a ClasSAE variant is the top scorer on every metric: ClasSAE L^1 on both cosine similarities, ClasSAE Hard on precision and F1, and the soft-gradient ClasSAE on recall. Standard is a surprisingly strong baseline, in particular on median CS, where it exceeds the soft-gradient ClasSAE, but trails all three ClasSAE variants on every other metric. Most notably, ClasSAE Shuffle collapses to near-zero across the board, confirming that the gain over baselines comes from real class structure and not merely from having a guidance signal at all.

An Out-of-the-Box Classifier Beyond annotation, the same construction gives ClasSAE a classifier for free: because M is trained to agree with the encoder's own selections, the pair (SAE, M) can be read directly as a ready-to-use classifier, with no additional training or fitting required.

Table 4: **Targeted probe perturbation.** TPP at $k \in \{20, 200, 600\}$ and under the PMI-graded edit (TPP PMI); Edit Strength is the mean L^1 norm of the TPP PMI edit. ClasSAE Hard leads at $k \geq 200$ and ClasSAE L^1 under TPP PMI, with the two smallest edit strengths.

Architecture	TPP 20 ↑	TPP 200 ↑	TPP 600 ↑	TPP PMI ↑	Edit Strength ↓
BatchTopK	$0.450_{\pm 0.003}$	$0.266_{\pm 0.011}$	$0.051_{\pm 0.003}$	$0.277_{\pm 0.016}$	$70.850_{\pm 4.841}$
JumpReLU	$0.102_{\pm 0.009}$	$0.463_{\pm 0.001}$	$0.281_{\pm 0.006}$	$0.453_{\pm 0.003}$	$87.082_{\pm 5.024}$
Matryoshka	$0.431_{\pm 0.008}$	$0.113_{\pm 0.009}$	$0.015_{\pm 0.003}$	$0.104_{\pm 0.004}$	$162.848_{\pm 2.378}$
Standard	$0.428_{\pm 0.003}$	$0.465_{\pm 0.001}$	$0.432_{\pm 0.003}$	$0.430_{\pm 0.001}$	$62.379_{\pm 0.196}$
Top-AFA	$0.174_{\pm 0.014}$	$0.464_{\pm 0.004}$	$0.356_{\pm 0.019}$	$0.012_{\pm 0.002}$	$31.282_{\pm 3.376}$
TopK	$0.423_{\pm 0.010}$	$0.304_{\pm 0.007}$	$0.067_{\pm 0.008}$	$0.309_{\pm 0.010}$	$124.047_{\pm 1.017}$
ClasSAE Shuffle	$\mathbf{0.453}_{\pm 0.004}$	$0.272_{\pm 0.009}$	$0.052_{\pm 0.004}$	$0.001_{\pm 0.001}$	$36.936_{\pm 0.619}$
ClasSAE	$0.082_{\pm 0.009}$	$0.471_{\pm 0.004}$	$0.513_{\pm 0.002}$	$0.462_{\pm 0.002}$	$33.816_{\pm 0.625}$
ClasSAE Hard	$0.064_{\pm 0.004}$	$\mathbf{0.479}_{\pm 0.002}$	$\mathbf{0.517}_{\pm 0.003}$	$0.469_{\pm 0.001}$	$\mathbf{23.470}_{\pm 0.393}$
ClasSAE L^1	$0.011_{\pm 0.001}$	$0.394_{\pm 0.003}$	$0.474_{\pm 0.000}$	$\mathbf{0.475}_{\pm 0.001}$	$23.842_{\pm 0.121}$

Prediction follows the same construction as the agreement loss itself: encode x, compute $\pi(x)$, and score each class by $\pi(x)^\top M$ (or $\pi(x)^\top \hat{M}$ for the post-hoc baselines); the predicted class is the argmax over this score. No separate classifier is fit at any point. With this construction we can report standard classification metrics in Table 3.

ClasSAE L^1 is the strongest predictor here, and the other two ClasSAE variants trained with true labels follow closely. The only real competitor is Standard, unsurprising given its strong showing in Table 2. More interesting is that Standard actually outperforms ClasSAE (the soft-gradient variant) on accuracy and F1, even though it has lower FVE in Table 1. We read this as evidence that reconstruction quality and class separability are, at some point, in direct competition: the soft-gradient ClasSAE trades some classification performance for the best FVE among the ClasSAE variants, while Standard's comparatively poor reconstruction leaves more of its representational capacity free to encode class structure.

Editing What You Named If the agreement loss genuinely reshapes the encoder rather than just labeling it, intervening on a target class should be easier and leak less into others. We test this with the standard TPP benchmark (Karvonen et al., 2025), at $k \in \{20, 200, 600\}$ selected features per class, and a custom variant **TPP PMI** described in Section D; a further unlearning task on a joint classifier, using the same intervention, is reported in Section E.

At TPP 20, ClasSAE Shuffle scores best, but this is an artifact of the top-k sensitivity we acknowledge in Section D rather than a real advantage: ClasSAE variants simply need more features to represent a class than an unsupervised encoder does. At TPP 600 and TPP PMI, all three ClasSAE variants lead, and at TPP 200 ClasSAE and ClasSAE Hard lead. ClasSAE Hard and ClasSAE L^1 achieve this with the lowest edit strength of all architectures, and the soft-gradient ClasSAE with lower edit strength than every baseline except Top-AFA, whose TPP PMI score is near zero. This indicates a genuinely better-separated encoder.

5 Conclusions

We introduced ClasSAE, an SAE that automates feature-class association and guides the encoder toward class-separable representations, with three sparsity variants. We derived theoretical properties of this construction, motivating both the architecture and our evaluation design. Empirically, ClasSAE delivers a strong suite of downstream capabilities, including out-of-the-box classification and precise, low-strength intervention. It leads on annotation faithfulness, classification, and TPP at larger k and under PMI-graded editing, at the cost of somewhat lower reconstruction quality and weaker results at small TPP k. Across these trade-offs, we recommend ClasSAE Hard as the default variant, as it offers the best balance overall.

Limitations SoftTopK is a major computational bottleneck, limiting scaling to larger dictionaries and to very large class counts, though 2^{14} classes is already an unlikely regime in practice.

Reproducibility Statement

Source code, configuration files, data preprocessing, and evaluation scripts will be released with the camera-ready version. The soft top-k operator is defined in Equation (2); the full objective in Equation (6); all hyperparameters, hardware, and wall-clock times in Section C. Every reported number is a mean over five seeds with its standard deviation. The proof of Proposition 1 is in Section A, and the derivation of Proposition 2 is in Section B.

Ethics Statement

This work contributes tooling for interpreting and steering vision and language models, which we take to be net-positive for safety and auditing. Interpretability methods can in principle assist an adversary in locating exploitable model behavior; we judge the defensive value to dominate here. ClasSAE attaches class labels to dictionary entries and therefore inherits any bias present in the label taxonomy: a feature annotated with a demographic class should be read as a statement about the label distribution, not about the concept. All datasets used are public and standard.

AI Use Statement

We used generative AI tools for language editing and for refining the presentation of text we had written ourselves. We did not use generative AI tools for research ideation, for deriving or checking the results in Section 3, or for generating experimental results. AI-assisted code completion was used during implementation; all such code was read, tested, and verified by the authors. We have reviewed all AI-assisted work and take responsibility for the final content of this paper.

References

Hiba Ahsan and Byron Wallace. Can saes reveal and mitigate racial biases of llms in healthcare? In *International Conference on Learning Representations*, volume 2026, pp. 4154–4172, 2026.

Tomer Ashuach, Dana Arad, Aaron Mueller, Martin Tutek, and Yonatan Belinkov. Crisp: Persistent concept unlearning via sparse autoencoders. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1806–1825, 2026.

Usha Bhalla, Alex Oesterling, Claudio Mayrink Verdun, Himabindu Lakkaraju, and Flavio P. Calmon. Temporal sparse autoencoders: Leveraging the sequential nature of language for interpretability. In *International Conference on Learning Representations (ICLR)*, 2026.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. In *NeurIPS 2024 Workshop on Scientific Methods for Understanding Deep Learning*, 2024.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=m25T5rAy43.

Kirill Bykov, Laura Kopf, Shinichi Nakajima, Marius Kloft, and Marina M.-C. Höhne. Labeling neural representations with inverse recognition. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Enrico Cassano, Riccardo Renzulli, Rayyan Ahmed, Marco Grangetto, and Stephan Alaniz. Look but don't touch with sparse autoencoders for unlearning in diffusion models. *arXiv preprint arXiv:2606.31699*, 2026a.

Enrico Cassano, Riccardo Renzulli, Marco Nurisso, Mirko Zaffaroni, Alan Perotti, and Marco Grangetto. SAEmnesia: Erasing concepts in diffusion models with supervised sparse autoencoders. In *Forty-third International Conference on Machine Learning*, 2026b.

David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, Satvik Golechha, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint arXiv:2409.14507*, 2025.

Ikhyun Cho and Julia Hockenmaier. Toward efficient sparse autoencoder-guided steering for improved in-context learning in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 28949–28961, 2025.

Bartosz Cywiński and Kamil Deja. SAeUron: Interpretable concept unlearning in diffusion models with sparse autoencoders. *International Conference on Machine Learning (ICML)*, 2025.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. IEEE, 2009.

Joshua Engels, Logan Riggs, and Max Tegmark. Decomposing the dark matter of sparse autoencoders. *Transactions on Machine Learning Research (TMLR)*, 2025.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, Pietro Lió, and Mateja Jamnik. Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Thomas Fel, Ekdeep Singh Lubana, Jacob S. Prince, Matthew Kowal, Victor Boutin, Isabel Papadimitriou, Binxu Wang, Martin Wattenberg, Demba Ba, and Talia Konkle. Archetypal SAE: Adaptive and stable dictionary learning for concept extraction in large vision models. In *International Conference on Machine Learning (ICML)*, 2025.

Vegard Flovik. Salve: Sparse autoencoder-latent vector editing for mechanistic control of neural networks. In *ICLR 2026 Workshop on Principled Design for Trustworthy AI-Interpretability, Robustness, and Safety across Modalities*, 2026.

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

Amirata Ghorbani, James Wexler, James Zou, and Been Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Yoav Gur-Arieh, Roy Mayan, Chen Agassy, Atticus Geiger, and Mor Geva. Enhancing automated interpretability with output-centric feature descriptions. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5757–5778, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.288. URL https://aclanthology.org/2025.acl-long.288/.

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

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Isaac Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum Stuart Mcdougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. In Aarti Singh, Maryam Fazel, Daniel

Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29223–29264. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/karvonen25a.html.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *International Conference on Machine Learning (ICML)*, 2018.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning (ICML)*, 2020.

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

Sewoong Lee, Adam Davies, Marc E. Canby, and Julia Hockenmaier. Evaluating and designing sparse autoencoders by approximating quasi-orthogonality. *Conference on Language Modeling (COLM)*, 2025. arXiv:2503.24277.

Ekdeep Singh Lubana, Can Rager, Sai Sumedh R. Hindupur, Valérie Costa, Greta Tuckute, Oam Patel, Sonia Krishna Murthy, Thomas Fel, Daniel Wurgaft, Eric J. Bigelow, Johnny Lin, Demba Ba, Martin Wattenberg, Fernanda Viegas, Melanie Weber, and Aaron Mueller. Priors in time: Missing inductive biases for language model interpretability. In *International Conference on Learning Representations (ICLR)*, 2026.

Alireza Makhzani and Brendan Frey. k-sparse autoencoders. *International Conference on Learning Representations (ICLR)*, 2014.

Gonçalo Paulo and Nora Belrose. Sparse autoencoders trained on the same data learn different features. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=EjInprGpk9.

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

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*, 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.

Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016. URL http://arxiv.org/abs/1610.02391.

Lee Sharkey, Dan Braun, and Beren Millidge. Taking features out of superposition with sparse autoencoders, 2022. AI Alignment Forum.

Łukasz Struski, Michał B. Bednarczyk, Igor T. Podolak, and Jacek Tabor. Lapsum - one method to differentiate them all: Ranking, sorting and top-k selection. In *The International Conference on Machine Learning (ICML) 2025*, 2025.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jermyn, et al. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.

Yujia Xie, Hanjun Dai, Minshuo Chen, Bo Dai, Tuo Zhao, Hongyuan Zha, Wei Wei, and Tomas Pfister. Differentiable top-k with optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Vladimir Zaigrajew, Hubert Baniecki, and Przemyslaw Biecek. Interpreting CLIP with hierarchical sparse autoencoders. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=5MQQsenQBm.

Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *European Conference on Computer Vision (ECCV)*, 2018.

A Proof of Proposition 1

Proof. By Equation (2), $M_i = \mathrm{SoftTopK}(\Lambda_i, k_i, \alpha_M)$ satisfies $\sum_c M_{ic} = k_i$ and $M_{ic} \in (0,1)$, and as Λ_i ranges over $\mathbb{R}^C$ the row M_i ranges over the whole relative interior of $\{m \in [0,1]^C : \sum_c m_c = k_i\}$. Since $k = K_{\mathrm{tot}}\,\mathrm{softmax}(\kappa)$ ranges over the relative interior of the simplex scaled by K_{tot}, the pair (κ, Λ) reaches exactly the relative interior of

$$\mathcal{M} = \Big\{ M \in [0,1]^{d\times C} \;:\; \textstyle\sum_{i,c} M_{ic} = K_{\mathrm{tot}} \Big\},$$

whose closure is $\mathcal{M}$. The objective $M \mapsto \langle M, \bar{A}\rangle$ is linear and $\mathcal{M}$ is a compact polytope (a box intersected with one hyperplane), so the maximum is attained at a vertex. Vertices of $\mathcal{M}$ have all coordinates in $\{0,1\}$ except at most one, so a maximizer is the indicator of the K_{tot} largest entries of $\bar{A}$ (with an arbitrary tie-break, and one fractional entry if K_{tot} is not an integer). Writing ν for the $(K_{\mathrm{tot}+1})$-th largest value of $\bar{A}$ gives $M^\star_{ic} = \mathbf{1}[\bar{A}_{ic} > \nu]$ and hence $k^\star_i = \sum_c M^\star_{ic} = |\{c : \bar{A}_{ic} > \nu\}|$. □

Two caveats. The statement holds for a *fixed* encoder; in joint training $\bar{A}$ moves, and the coupled problem is not linear. It also holds only for the base objective; the contrastive form is not linear in M, and softens the vertex solution without changing the water-filling reading.

B Proof of Proposition 2

Proof. **Step 1: the hard-limit correspondence.** As $\alpha_p \to 0$, $p_i(x) \to A_i(x) = \mathbf{1}[i \in S(x)]$ pointwise, since SoftTopK converges to the indicator of the selected support (Equation (2)). With support size held constant, $\hat{k}(x) \equiv \bar{k}$, so

$$\pi_i(x) = \frac{p_i(x)}{\hat{k}(x)} \;\longrightarrow\; \frac{A_i(x)}{\bar{k}}.$$

Step 2: $\bar{A}$ reduces to a rescaled conditional firing rate. By definition, $\bar{A}_{ic} = \mathbb{E}_{x\sim c}[\pi_i(x)]$. Taking the limit of Step 1 inside the expectation,

$$\bar{A}_{ic} \;\longrightarrow\; \frac{1}{\bar{k}}\,\mathbb{E}_{x\sim c}[A_i(x)] = \frac{P(A_i{=}1 \mid c)}{\bar{k}}.$$

Step 3: recovering PMI. By definition, $\mathrm{PMI}(i,c) = \log P(A_i{=}1 \mid c) - \log P(A_i{=}1)$. Substituting Step 2,

$$\log \bar{A}_{ic} = \log P(A_i{=}1 \mid c) - \log \bar{k} = \mathrm{PMI}(i,c) + \log P(A_i{=}1) - \log \bar{k},$$

so

$$\mathrm{PMI}(i,c) = \log \bar{A}_{ic} - \log P(A_i{=}1) + \log \bar{k}. \qquad (*)$$

Step 4: substituting the water-filling threshold. By Proposition 1, $M^\star_{ic} = \mathbf{1}[\bar{A}_{ic} > \nu]$ for a single global threshold ν. Equivalently, $\log \bar{A}_{ic} > \log \nu$. Using $(*)$ to rewrite $\log \bar{A}_{ic}$ in terms of $\mathrm{PMI}(i,c)$,

$$\begin{aligned} \log \bar{A}_{ic} > \log \nu \;&\Longleftrightarrow \\ \mathrm{PMI}(i,c) + \log P(A_i{=}1) - \log \bar{k} > \log \nu \;&\Longleftrightarrow \\ \mathrm{PMI}(i,c) > \log(\nu \bar{k}) - \log P(A_i{=}1). \end{aligned}$$

Setting $\nu_i := \log(\nu\bar{k}) - \log P(A_i{=}1)$ gives exactly

$$M^\star_{ic} \;\longrightarrow\; \mathbf{1}\big[\mathrm{PMI}(i,c) > \nu_i\big],$$

which is equation 7. Since ν is shared across all features while $P(A_i{=}1)$ is feature-specific, ν_i is a per-feature calibration of the single global threshold ν, decreasing in i's marginal firing rate. □

Departure from the idealized limit This correspondence assumes constant support size, $\hat{k}(x) \equiv \bar{k}$, which holds for plain per-sample top-k but not for BatchTopK, where $\hat{k}(x)$ is itself a per-sample random variable. Writing $m_c := \mathbb{E}_{x\sim c}[\hat{k}(x)]$ and expanding $1/\hat{k}(x)$ to first order around m_c,

$$\bar{A}_{ic} \approx \frac{P(A_i{=}1 \mid c)}{m_c} - m_c^{-2}\,\mathrm{Cov}_{x\sim c}\big(A_i(x), \hat{k}(x)\big),$$

so under BatchTopK the correspondence holds only approximately, with a residual governed by $\mathrm{Var}(\hat{k} \mid c)$ and by any drift of m_c from $\bar{k}$ across classes. This applies equally to variant (iii): although the L^1-relaxed encoder has no sparse reconstruction, $A(x)$ and $\hat{k}(x)$ are defined through the same agreement-side $p(x)$ used by every variant (Section 3), so the correspondence and its BatchTopK-style correction apply here as well, with m_c taken over this variant's own $\hat{k}(x)$.

C Hyperparameters

Compute All architectures except ClasSAE trained in under 10 minutes; ClasSAE variants took approximately 1 hour each, all on a single NVIDIA GH200.

Shared across all architectures Epochs 50, Learning rate 6×10^{-4}, dictionary size $d = 16384$, activation dimension $n = 768$, auxiliary-loss coefficient $\gamma = 1/32$, LR decay start at step 5000, 1000 warmup steps, threshold start at step 1000, dead-feature threshold 2×10^6 steps, and no $\hat{k}$-annealing.

ClasSAE variants (Shuffle, ClasSAE, ClasSAE Hard, ClasSAE L^1) Association budget $\rho = 40$, $C = 1000$ classes, contrastive temperature $\tau = 1$, agreement-loss weight $\mu = 0.2$, SoftTopK temperature $\alpha_p = \alpha_M = 0.001$, τ-annealing turned off, α annealing turned off. Target $\bar{k} = 380$ for Shuffle, ClasSAE, and ClasSAE Hard; ClasSAE L^1 instead uses an L^1 coefficient of 0.05 in place of a hard target, with no L^1-annealing schedule. Threshold EMA $\beta = 0.999$.

BatchTopK Target $\bar{k} = 380$; threshold EMA $\beta = 0.999$.

JumpReLU Target $L_0 = 380$, bandwidth 0.001, sparsity penalty coefficient 1, sparsity warmup over 300 steps, threshold EMA $\beta = 0.999$.

Matryoshka Target $\bar{k} = 380$; group fractions $(1/64,\ 1/64,\ 1/32,\ 1/16,\ 1/8,\ 1/4,\ 1/2)$, uniform group weighting; threshold EMA $\beta = 0.999$.

Standard L^1 coefficient 0.025, sparsity warmup over 300 steps, no feature resampling; threshold EMA $\beta = 0.999$.

Top-AFA L^1 coefficient 0, AFA coefficient 0.0625, auxiliary penalty 0.03125, no auxiliary top-k.

TopK Target $\bar{k} = 380$; threshold EMA $\beta = 0.999$.

D Targeted Probe Perturbation Setup

TPP PMI TPP's top-k selection makes cross-architecture comparison sensitive to the choice of k, since an architecture can concentrate its signal into few features and win at small k but lose at large k or vice versa; this is exactly what we observe in Table 4. We therefore also grade intervention strength continuously rather than thresholding at top-k, following Proposition 2: given activation z,

$$z' = z \odot (1 - \text{strength}), \qquad \text{strength} = \text{clamp}_{\leq 1}\left(\frac{z}{q_{0.9}(z)}\right) \odot \frac{\text{clamp}_{>0}(\text{PMI}(\cdot, c))}{q_{0.99}\big(\text{clamp}_{>0}(\text{PMI}(\cdot, c))\big)} \cdot \eta, \tag{8}$$

with $q_{0.9}(z)$ a per-feature activation quantile, $q_{0.99}(\cdot)$ the 99th percentile used to normalize the clamped PMI scores, and η a strength multiplier, so common low-activation features are left largely untouched while strongly firing, class-informative features are targeted. **TPP PMI** is TPP evaluated with this edit in place of top-k selection; **Edit Strength** is the mean L^1 norm of the strength vector.

TPP setup and parameters We evaluate 50 concepts (sampled at random rather than fixed by index, but the same for each architecture), with a balanced negative-to-positive ratio of 1.0 for each concept's binary partition. Probes are trained for 20 epochs, learning rate 10^{-3}, weight decay 10^{-4}, batch size 256.

Asociation budget $\rho = 40$, edit-strength scale 5.0 (up from a default of 1.0), PMI upper percentile 99.0, and activation-scale percentile 90.0. For ClasSAE variants, we use the model's own M and native selection weights directly rather than an empirical PMI pass; all other architectures use the empirical PMI pass by default.

Data are processed in batches of 4096 with chunk size 2048 and 4 worker processes; we use the full train and test activation sets (no cap on the number of examples) and seed 0.

We use the same setting for the Unlearning Benchmark in the next section.

E Custom Unlearning Benchmark

Table 5: **Joint-classifier unlearning.** Graded-vs-shuffled forget-accuracy drop, retain-accuracy drop, and edit strength (mean L^1 norm); the two drops are measured relative to baseline classifier accuracy. ClasSAE variants dominate on forget drop while keeping edit strength small.

Architecture	Graded vs. Shuffled Forget Drop ↑	Retain Drop ↓	Edit Strength ↓
BatchTopK	-0.284$_{\pm 0.010}$	0.075$_{\pm 0.009}$	70.850$_{\pm 4.841}$
JumpReLU	0.170$_{\pm 0.017}$	0.034$_{\pm 0.003}$	87.082$_{\pm 5.024}$
Matryoshka	0.000$_{\pm 0.000}$	0.421$_{\pm 0.014}$	162.848$_{\pm 2.378}$
Standard	0.224$_{\pm 0.046}$	0.055$_{\pm 0.000}$	62.379$_{\pm 0.196}$
Top-AFA	-0.442$_{\pm 0.032}$	**0.001**$_{\pm 0.000}$	31.282$_{\pm 3.376}$
TopK	0.000$_{\pm 0.000}$	0.199$_{\pm 0.006}$	124.047$_{\pm 1.017}$
ClasSAE Shuffle	0.016$_{\pm 0.014}$	0.032$_{\pm 0.001}$	36.936$_{\pm 0.619}$
ClasSAE	**0.816**$_{\pm 0.008}$	0.023$_{\pm 0.002}$	33.816$_{\pm 0.625}$
ClasSAE Hard	**0.816**$_{\pm 0.009}$	0.011$_{\pm 0.002}$	**23.470**$_{\pm 0.393}$
ClasSAE L^1	0.803$_{\pm 0.001}$	0.002$_{\pm 0.000}$	23.842$_{\pm 0.121}$

As a complement to the targeted probe perturbation results in Section 4, we evaluate the same PMI-graded intervention on a joint multi-class classifier rather than independent per-class probes, testing whether selective ablation still works when classes must be distinguished by a single shared decision boundary rather than C separate ones.

We train a linear classifier on SAE-reconstructed embeddings and record its baseline forget and retain accuracy. We then apply two edits: the graded intervention of Section 4, and the same intervention with PMI scores shuffled among the features that had nonzero PMI for the target class, isolating whether the intervention's quality comes from targeting the right subset of features or from the fine-grained ranking within it. Table 5 reports the drop in forget accuracy under the graded edit relative to baseline, minus the same drop under the shuffled edit (**Graded vs. Shuffled Forget Drop**); the drop in retain accuracy relative to baseline (**Retain Drop**); and **Edit Strength**, the mean L^1 norm of the strength vector, as in Section 4.

On the forget-drop metric, all three ClasSAE variants trained with true labels dominate by a wide margin. Here, shuffling PMI scores within the selected subset sharply degrades the edit, showing that ClasSAE's selected features are not interchangeable: the fine-grained ranking within the subset, not just membership in it, is what drives the intervention. Baselines like JumpReLU tell the opposite story: they remain effective even after shuffling, meaning their selected features are largely interchangeable and carry little differentiated class information individually. A negative value, as for BatchTopK and Top-AFA, means the shuffled edit achieved a *larger* forget-accuracy drop than the graded one. On retain drop, ClasSAE variants are best apart from Top-AFA, which tops this column but is worst on forget drop. Edit strength stays small across all ClasSAE variants, consistent with Table 4. Together with the main-text results, this confirms that precise, low-strength intervention extends from independent probes to a joint classifier as well.

F Qualitative Example: Patch-Token Class Alignments

As a further demonstration of ClasSAE's applicability beyond the CLIP/ImageNet setting used throughout the paper, we train ClasSAE on patch tokens from DINOv3 (`facebook/dinov3-vitb16-pretrain-lvd1689m`), assigning every patch token the class label of its parent image rather than any finer-grained, per-patch segmentation label. Despite this deliberately naive labeling scheme, in which a single whole-image class is spread uniformly across all of an image's patch tokens, we find that ClasSAE still learns individual features that fire specifically on the image region corresponding to their claimed class.

Figure 2 shows two examples on an image containing both a Siamese cat and an American Staffordshire terrier. The top row shows features claimed by "Siamese cat, Siamese," each activating on the cat rather than the dog, with sharper spatial localization for the lower-budget, more specialist feature (feat. 1277, $k_i = 1.0$) than for the higher-budget, more generalist one (feat. 15, $k_i = 8.0$). The bottom row shows the analogous pattern for features claimed by "American Staffordshire terrier, Staffordshire terrier, American pit bull terrier, pit bull terrier," localizing on the dog. This is a further example of how ClasSAE can be used in practice, extracting spatially meaningful, class-aligned features even from a naive, image-level labeling scheme with no segmentation supervision.