SPARSE AUTOENCODERS CAN INDUCE LOCAL GEOMETRY FOR MITIGATING CATASTROPHIC FORGETTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Catastrophic forgetting remains a challenge in supervised continual learning, as adapting to new tasks causes representation drift and loss of previously learned information. This raises the question of which representational changes should be constrained while preserving plasticity. We therefore introduce *Sparse Autoencoder Geometric Anchoring (SAGA)*, which converts a Sparse Autoencoder (SAE) trained at each task boundary into local, low-rank constraints on representation drift. Replay examples specify where the constraints apply, while leading right singular directions of the encoder Jacobian specify which changes are penalized. The resulting geometry is input dependent and can be retained without keeping the SAE for subsequent tasks' training. We establish its local relation to latent code matching, define its spectral truncation error, and derive a conditional bound on frozen-readout drift. With replay and snapshot history, the method achieves the highest mean Class-IL accuracy and lowest mean forgetting among the evaluated ER, DER++, ER-ACE, and SGD baselines on three pretrained ViT-B/16 benchmarks, with accuracy gains of 12.15, 4.79, and 3.22 percentage points over the strongest conventional baseline on CIFAR-100, CUB-200, and ImageNet-R, respectively. From scratch results are more mixed, while rank ablations and geometry controls show that SAE-induced low-rank geometry offers useful stability-plasticity trade-offs.

1 INTRODUCTION

Catastrophic forgetting, i.e., the degradation of previously acquired knowledge during sequential training on new tasks, remains one of the key challenges in supervised continual learning Wang et al. (2024). Common mitigation strategies include rehearsal Rolnick et al. (2019), regularization and distillation Kirkpatrick et al. (2017); Li & Hoiem (2018), and hybrid approaches Buzzega et al. (2020). More recently, forgetting has increasingly been studied at the representation level, where learning subsequent tasks can alter feature-space geometry and move previously useful information into directions that are no longer accessible to the original readout Gu et al. (2023); Filus et al. (2026); Masip et al. (2026). This motivates preserving task-relevant representation subspaces, for example those derived from previous task heads Yamashita et al. (2026) or representations Saha et al. (2021). However, these approaches typically define global subspaces, leaving open whether local geometry could provide selective control of representation drift.

Sparse autoencoders (SAEs) map dense representations to sparse feature codes Huben et al. (2024); Bereska & Gavves (2024); Filus & Pokuciński (2026). In continual learning, they have been used to analyze changes in visual representations Filus et al. (2026); Masip et al. (2026) and to regularize sparse activations during language-model adaptation Zhang et al. (2026); Ning et al. (2026). We ask a different question: *can a sparse encoder specify which local changes in the original representation should be penalized, rather than only providing a latent code to match?*

To address this question, we introduce *SAGA* (*Sparse Autoencoder Geometric Anchoring*), which uses SAE encoder Jacobians to derive task-specific local geometry for representation anchoring. At each task boundary, a fresh SAE is fitted to the backbone representations of a replay buffer snapshot. Its encoder Jacobian defines a local quadratic geometry at each stored example. The snapshot specifies *where* to anchor the old representation, while the leading Jacobian singular directions and their weights specify *which changes* to penalize. Unlike a single global subspace, these constraints

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.

can differ across examples because the SAE's active features change with the input. Once these elements are computed, the SAE can be discarded. Full-stack SAGA combines this Jacobian anchor with two components. *History* keeps the anchors from all task boundaries, so older constraints are not overwritten by snapshots fitted to already-drifted representations. *Replay* adds cross-entropy on snapshot examples, supplying the classifier supervision that anchoring alone lacks in Class-IL.

The empirical benefit is clearest with pretrained representations. On CIFAR-100, CUB-200, and ImageNet-R, full-stack *SAGA* configuration achieves the highest mean accuracy and lowest mean forgetting among the evaluated conventional baselines. Against the corresponding latent-anchoring configuration, it has higher mean Class-IL accuracy in four out of five dataset/backbone settings, with CIFAR-100 being the exception. Rank ablations show that a small retained subspace can be sufficient, and geometry controls identify where SAE-derived constraints help relative to global alternatives. We distinguish these empirical findings from a universal optimality claim: the mathematical analysis states the coverage of old-readout sensitivity needed for a geometric anchor to control functional drift. Our contributions can be summarized in the following way:

- A task-boundary anchoring method that turns sparse encoders into input-dependent, low-rank quadratic constraints, separating spatial memory in replay examples from directional memory in Jacobian factors.
- An analysis of the relation to latent matching, spectral truncation, and frozen-readout drift, with an explicit distinction between geometric energy and the normalization of the implemented training losses.
- An empirical evaluation across pretrained and from scratch backbones, including replay/history ablations, rank compression, alternative geometries, and dictionary size, that characterizes both the benefits and the limits of SAE-derived anchoring.

2 Related work

Traditional research on catastrophic forgetting in supervised continual learning has primarily focused on classifier-level degradation and mitigation strategies such as replay, regularization, distillation, and architectural approaches Kirkpatrick et al. (2017); Li & Hoiem (2018); Parisi et al. (2019); Wang et al. (2024); Verwimp et al. (2024). More recently, attention has shifted toward representations, including analyses that distinguish information loss from loss of accessibility to an old readout Davari et al. (2022), and methods that explicitly preserve representation structure. LUCIR constrains feature directions Hou et al. (2019), PODNet distills spatially pooled representations Douillard et al. (2020), and Backward Feature Projection preserves linear separability of past tasks Gu et al. (2023). Related works preserve task-relevant subspaces defined from activations Saha et al. (2021) or classifier weights Yamashita et al. (2026). At the parameter level, OGD constrains updates using gradients of old outputs Farajtabar et al. (2020), while compressed quadratic constraints have been considered through sketched structural and Jacobian regularization Li et al. (2021); Heckel (2022). Thus, neither subspace nor Jacobian-based regularization is new in isolation. In contrast to these global or parameter-space constraints, our method uses the Jacobian of a task-anchored SAE encoder with respect to the backbone representation to induce an input-dependent, low-rank geometry that adapts to each replay anchor.

The use of encoder Jacobians to capture local representation geometry has also been studied in earlier work on autoencoders. In particular, the Manifold Tangent Classifier extracts local directions from top singular vectors of a contractive autoencoder encoder Jacobian and encourages classifier invariance along them Rifai et al. (2011). Our objective is different: rather than interpreting these directions as manifold tangents and enforcing classifier invariance, we freeze SAE sensitivity factors at task boundaries and use them to constrain subsequent backbone representation drift in continual learning. We make no assumption that the selected TopK-Jacobian directions correspond to manifold tangents or individual semantic concepts. Our sparse encoders build on k-sparse autoencoders Makhzani & Frey (2013) and subsequent TopK SAE formulations Gao et al. (2025). Jacobian Sparse Autoencoders Farnik et al. (2025) address a complementary problem of learning sparse computational connections between model representations, rather than deriving continual-learning constraints from the local geometry of a frozen sparse encoder.

Recently, Sparse Autoencoders (SAEs), which map dense representations into sparse feature vectors Huben et al. (2024); Bereska & Gavves (2024); Filus & Pokuciński (2026), have been applied to continual learning both to analyze and mitigate forgetting. Diagnostic studies Filus et al. (2026); Masip et al. (2026) show that forgetting can reflect changes in feature accessibility and geometry rather than complete information loss, with a substantial component of representational drift being approximately linear. For mitigation in Low-Rank Adaptation (LoRA)-based continual learning for LLMs, SAE-FD Zhang et al. (2026) uses directional feature matching and magnitude preservation, while SAE-guided activation regularization Ning et al. (2026) selectively protects SAE features. In contrast, we study supervised continual learning in vision and train *task-boundary SAEs* rather than relying on a shared pretrained SAE space. Instead of directly preserving SAE activations, we export their local sensitivity geometry as low-rank factors and constrain representation drift along its dominant directions, without retaining the SAE during subsequent training. Our *Latent* baseline therefore serves as a direct TopK-code matching ablation rather than a reproduction of these prior methods.

While subspace regularization, Jacobian-based geometry, and SAE-based preservation have each been explored in different contexts, our approach introduces *task-boundary, input-dependent SAE geometry* for continual representation anchoring. At each task boundary, the SAE is converted into local, low-rank sensitivity directions that constrain subsequent representation drift and can be retained without keeping the SAE itself. Comparisons with latent anchoring and random, PCA, and classifier SVD geometries isolate the role of the resulting SAE-induced geometry.

3 Using SAE-induced local geometry for forgetting mitigation

Spatial and directional anchors. Let s index a completed task and let $B_s = \{x_{s,i}\}_{i=1}^{n_s}$ be the reservoir-buffer snapshot stored at that boundary. It contains examples from the cumulative stream, not exclusively from task s. With backbone parameters θ_s, store $h_s(x) = f_{\theta_s}(x) \in \mathbb{R}^d$ and fit a fresh TopK SAE encoder E_s Makhzani & Frey (2013); Gao et al. (2025) to the snapshot representations. At each stored example, compute

$$J_s(x) = J_{E_s}(h_s(x)) = U_s(x)\Sigma_s(x)V_s(x)^\top, \qquad \delta_s(x;\theta) = f_\theta(x) - h_s(x). \tag{1}$$

Define the full pullback semimetric and its spectral truncation by

$$Q_s(h) = J_{E_s}(h)^\top J_{E_s}(h), \tag{2}$$

$$Q_{s,r}(x) = V_{s,r}(x)\Sigma_{s,r}(x)^2 V_{s,r}(x)^\top \succeq 0, \tag{3}$$

where $V_{s,r}$ contains the r retained right singular vectors. The per-example *geometric energy* and its snapshot average are

$$G_{s,r}(x;\theta) = \|\Sigma_{s,r}(x)V_{s,r}(x)^\top \delta_s(x;\theta)\|_2^2 = \delta_s(x;\theta)^\top Q_{s,r}(x)\delta_s(x;\theta), \tag{4}$$

$$\widehat{R}_{s,r}(\theta) = \frac{1}{|B_s|}\sum_{x\in B_s} G_{s,r}(x;\theta). \tag{5}$$

The snapshot specifies *where* the old representation is anchored and $Q_{s,r}(x)$ specifies *which local changes* are penalized. In Jacobian mode, the encoder is discarded after these factors have been computed; the cached representations and factors define the subsequent constraint. Appendix D.4 specifies the architecture and construction.

Training losses and snapshot history. For an anchor minibatch $\mathcal{M}_s \subset B_s$, the raw Jacobian loss averages over examples *and* retained directions, whereas the latent baseline sums latent-coordinate errors and averages over examples:

$$\mathcal{L}^{\text{impl}}_{\text{Jac},s} = \frac{1}{|\mathcal{M}_s| r}\sum_{x\in\mathcal{M}_s} G_{s,r}(x;\theta), \tag{6}$$

$$\mathcal{L}^{\text{impl}}_{\text{latent},s} = \frac{1}{|\mathcal{M}_s|}\sum_{x\in\mathcal{M}_s} \|E_s(f_\theta(x)) - E_s(h_s(x))\|_2^2. \tag{7}$$

Thus the raw training metric is $Q_{s,r}/r$, not $Q_{s,r}$. `Vanilla` retains the latest snapshot; `History` retains all completed-task snapshots. Let $\mathcal{H}_t$ be this retained index set when learning task t. An

independent minibatch is drawn from each snapshot and the default mean reduction is

$$\mathcal{L}^{\text{impl}}_{\text{anchor}} = \frac{1}{|\mathcal{H}_t|} \sum_{s \in \mathcal{H}_t} \mathcal{L}^{\text{impl}}_s. \tag{8}$$

`Replay` adds cross-entropy on the minibatch from the *latest frozen snapshot*, rather than drawing from the live reservoir. With λ the anchor coefficient and ρ the replay coefficient, the objective is

$$\mathcal{L} = \mathcal{L}_{\text{stream CE}} + \lambda \mathcal{L}^{\text{impl}}_{\text{anchor}} + \rho \mathcal{L}_{\text{snapshot CE}}. \tag{9}$$

Without `Replay`, $\rho = 0$. Before the first snapshot is created, only the stream loss is used. Stored images have deterministic preprocessing for anchor comparisons so that the old and current features refer to the same input view.

Relation to latent matching and rank truncation. For $E(h) = \text{TopK}(\text{ReLU}(Wh + b))$, TopK retains the K largest post-ReLU values in their original coordinates. If the positive active mask is unchanged along $h_0 + \tau\delta$, $0 \le \tau \le 1$, then

$$E(h_0 + \delta) - E(h_0) = J_E(h_0)\delta. \tag{10}$$

At full retained rank, the unnormalized Jacobian energy therefore equals latent displacement energy. The *implemented* raw losses satisfy $\mathcal{L}^{\text{impl}}_{\text{latent}} = r\mathcal{L}^{\text{impl}}_{\text{Jac}}$ within a fixed cell, so equal coefficients do not give equal local regularization strength. Our latent comparison concerns these implemented objectives, not a matched-strength isolation of support geometry.

For $Q = J^\top J$ and $Q_r = \sum_{i=1}^{r} \sigma_i^2 v_i v_i^\top$, spectral truncation is optimal in operator and Frobenius norms and gives

$$0 \le \delta^\top (Q - Q_r)\delta \le \sigma_{r+1}^2 \|\delta\|_2^2, \qquad \dim \ker Q_r \ge d - r. \tag{11}$$

Missing singular values are set to zero. The nullspace is local to an anchor: multiple examples or snapshots need not leave a shared subspace of that dimension. Support changes and truncation distinguish the geometric energies; the $1/r$ reduction additionally changes the scale of the training loss.

Functional relevance. Let $g_s : \mathbb{R}^d \to \mathbb{R}^{c_s}$ be a frozen old readout, $A_s = J_{g_s}(h_0)$, and let V_r, Σ_r retain positive singular values only. Writing $P_r = V_r V_r^\top$, define $\alpha_{s,r} = \|A_s V_r \Sigma_r^{-1}\|_{\text{op}}$ and $\beta_{s,r} = \|A_s(I - P_r)\|_{\text{op}}$. If J_{g_s} is L_g-Lipschitz on a convex neighborhood containing the displacement segment, then

$$\|g_s(h_0 + \delta) - g_s(h_0)\|_2 \le \alpha_{s,r}\sqrt{\delta^\top Q_r \delta} + \beta_{s,r}\|\delta\|_2 + \frac{L_g}{2}\|\delta\|_2^2. \tag{12}$$

This bound separates controlled geometric energy from old-readout sensitivity outside the protected subspace. It is a conditional frozen-readout statement, not a guarantee for the changing Class-IL classifier. We do not estimate $\beta_{s,r}$ in the reported experiments. Appendix A provides the proofs, support-boundary example, and further consequences.

4 Experimental setup

Datasets. CIFAR-100 (100 classes, split into 10 tasks of 10 classes each; used both at native 32×32 resolution with a from scratch backbone, and upsampled to 224×224 with a pretrained backbone). CUB-200 (200 fine-grained bird species, 10 tasks of 20 classes each, 224×224). ImageNet-R (200 classes – renditions/art styles of ImageNet classes, 10 tasks of 20 classes each, 224×224). TinyImageNet (200 classes, split into 10 tasks of 20 classes each). For the from scratch regime we use CIFAR-100 and TinyImageNet, which provide 500 training images per class, rather than CUB-200 and ImageNet-R (30 and 120 images per class), as training on such limited data is not possible. more details are described in Appendix D.

Backbones. ViT-B/16, pretrained on ImageNet-21k. ResNet-18, trained entirely from scratch. We include more details on the hyperparameters in Appendix D.2

Scenario and metrics. All experiments use the class-incremental learning (Class-IL) scenario, where the task identity is not available at test time. Task-IL scenario (task identity given at test time) is reported alongside as a secondary, less demanding diagnostic in Appendix B. We implement our proposed methods within the Mammoth continual learning framework[1] Buzzega et al. (2020); Boschini et al. (2022) and use it for all experiments. The primary metrics are final mean accuracy (accuracy on all classes seen so far, averaged over tasks, after training on the last task) and forgetting (mean drop from each task's peak accuracy to its accuracy at the end of training), computed using Mammoth's standard evaluation protocol.

Protocol. A replay buffer of size 500 (5120 for TinyImagenet) is used throughout for every rehearsal-based method (ours and all four baselines). Every number reported as "final" anywhere in this paper is the mean $\pm$ standard deviation over three seeds ($\{0, 1, 2\}$) on the true test set; hyperparameter selection (Section D.2) is always performed separately on a held-out validation split and never touches the test set used for the numbers reported here.

Anchor snapshots and memory. The live reservoir contains 500 examples, but `History` retains an additional frozen task-boundary snapshot after each completed task. Each snapshot stores its then-current reservoir images together with old representations and the corresponding anchor factors. Thus History has auxiliary memory and anchor-computation costs that grow with the number of completed tasks, even though the live replay reservoir remains fixed at 500 examples; we therefore do not claim a strictly memory-matched comparison to ER, DER++, or ER-ACE. The default mean History reduction in Eq. 8 keeps the nominal anchor-loss scale approximately independent of the number of retained snapshots.

Evaluation plan. Our main experimental evaluation can be organized into four parts: **(1)** Ablation regarding the design choices for the task-boundary SAE-based anchoring and a comparison between SAE-Jacobian anchoring and direct SAE-latent anchoring. In the ablation we include combinations of `Vanilla Latent`/`Vanilla Jacobian`, `History` and `Replay` configurations on a full experimental stack (Section 5.1); **(2)** Comparison of the Jacobian-based anchoring (`SAGA`) with established continual learning baselines, including SGD (no forgetting mitigation), Experience Replay (ER) Chaudhry et al. (2019); Rolnick et al. (2019), Dark Experience Replay with Knowledge Distillation (DER++) Buzzega et al. (2020), and Experience Replay with Asymmetric Cross-Entropy (ER-ACE) Caccia et al. (2022) on a full experimental stack (for further experimental details see Appendix D.3, and for the results see Section 5.2); **(3)** Analysis of the effect of the retained Jacobian rank (r) to assess performance in low-rank anchoring regimes on example configurations (Section 5.3); **(4)** Comparison of SAE-induced geometry with alternative reference subspaces used for anchoring derived from random directions, Principal Component Analysis (PCA), and classifier SVD on example experimental configuration (Section 5.4). We also perform the analysis of the effect of SAE dictionary size in Appendix C, ranging from undercomplete to strongly overcomplete configurations, to determine whether overcompleteness is necessary for effective anchoring. For computational efficiency, we use undercomplete SAEs in the main experiments, while our dictionary-size ablation in Appendix C shows that overcomplete SAEs achieve comparable anchoring performance and can therefore be used when interpretability is also desired.

5 Experimental results

5.1 Comparison of Jacobian-based anchoring with Latent-based anchoring with SAEs

In this section, we compare `Jacobian` and direct `Latent` anchoring separately for **pretrained** backbones (model: ViT-B/16, datasets: CIFAR-100, CUB-200, ImageNet-R) and backbones trained **from scratch** (model: ResNet-18, datasets: CIFAR-100, TinyImageNet), along with the ablation study of the design choices for both these methods. Tables 1 and 2 report the Class-IL accuracy and forgetting results for the pretrained and from scratch cases respectively. Task-IL results are provided in Appendix B.

The clearest ablation result is the benefit of combining replay with historical anchors. For both anchor types and all five settings, the full `Replay + History` configuration improves both mean

[1] `https://github.com/aimagelab/mammoth`

Table 1: SAE anchoring ablations for Class-IL scenario with **pretrained** ViT-B/16. Results are mean ± std over three seeds. Higher accuracy (↑) and lower forgetting (↓) are better.

	CIFAR-100		CUB-200		ImageNet-R	
Configuration	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Vanilla Latent	31.58 ± 3.19	72.95 ± 3.64	37.89 ± 0.79	51.36 ± 0.70	33.76 ± 0.36	64.69 ± 0.25
Vanilla Jacobian	27.57 ± 0.95	78.69 ± 1.13	33.03 ± 0.80	60.94 ± 0.46	28.15 ± 1.09	71.86 ± 1.28
Latent + History	56.65 ± 0.46	45.77 ± 0.54	61.21 ± 1.22	27.77 ± 1.26	48.61 ± 0.29	49.33 ± 0.24
Jacobian + History	51.37 ± 1.89	52.12 ± 2.09	59.53 ± 0.37	31.79 ± 0.38	45.06 ± 1.47	53.00 ± 1.45
Latent + Replay	70.96 ± 0.21	25.82 ± 0.84	64.45 ± 1.64	20.96 ± 1.53	53.78 ± 0.19	42.32 ± 0.36
Jacobian + Replay	72.31 ± 0.91	28.49 ± 1.15	66.85 ± 1.77	23.04 ± 1.79	53.67 ± 0.78	43.38 ± 0.98
Latent + Replay + History	$\mathbf{78.48 \pm 0.73}$	$\mathbf{19.83 \pm 1.04}$	72.78 ± 0.96	$\mathbf{14.04 \pm 0.92}$	58.96 ± 1.18	37.55 ± 1.04
SAGA (Jacobian + Replay + History)	76.01 ± 0.46	24.24 ± 0.43	$\mathbf{73.68 \pm 0.79}$	15.38 ± 0.92	$\mathbf{61.58 \pm 0.51}$	$\mathbf{34.48 \pm 0.40}$

Class-IL accuracy and mean forgetting over the corresponding `Replay`-only configuration. History alone helps substantially on pretrained representations, but is less reliable from scratch. Thus the useful setting is geometric anchoring together with replay and retained snapshots, rather than either standalone anchor.

The comparison between the full latent and Jacobian configurations is more specific. Jacobian anchoring improves mean accuracy in four of five settings. On from scratch CIFAR-100, accuracy rises from 21.44% to 28.53% and forgetting falls from 48.90 to 38.46; both metrics also improve on TinyImageNet and pretrained ImageNet-R. CUB-200 trades slightly higher accuracy for higher forgetting, while latent anchoring is better on CIFAR-100. These outcomes characterize the implemented methods; the differing loss reductions in Eqs. 6 and equation 7 prevent attributing every difference solely to the choice of local geometry.

Table 2: SAE anchoring ablations for Class-IL scenario with ResNet-18 trained **from scratch**. Results are mean ± std over three seeds. Higher accuracy (↑) and lower forgetting (↓) are better.

	CIFAR-100		TinyImageNet	
Configuration	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Vanilla Latent	8.03 ± 0.74	59.32 ± 3.53	6.64 ± 0.19	57.79 ± 3.37
Vanilla Jacobian	8.16 ± 1.20	69.50 ± 5.43	7.32 ± 0.45	61.78 ± 4.50
Latent + History	7.90 ± 0.34	58.14 ± 4.18	6.44 ± 0.08	58.85 ± 3.26
Jacobian + History	8.20 ± 0.52	66.50 ± 7.42	5.89 ± 0.62	56.40 ± 5.09
Latent + Replay	15.78 ± 0.29	55.26 ± 2.07	17.79 ± 0.83	38.55 ± 4.37
Jacobian + Replay	17.53 ± 1.87	55.24 ± 3.69	16.07 ± 3.46	40.24 ± 5.92
Latent + Replay + History	21.44 ± 1.12	48.90 ± 0.96	25.45 ± 1.51	17.10 ± 0.62
Jacobian + Replay + History	$\mathbf{28.53 \pm 0.52}$	$\mathbf{38.46 \pm 2.64}$	$\mathbf{28.27 \pm 4.43}$	$\mathbf{12.84 \pm 2.24}$

5.2 Comparison between Jacobian-based SAE anchoring and common forgetting mitigation methods

In this section, we compare `Jacobian` anchoring with common continual learning strategies – SGD (no forgetting mitigation), ER, ER-ACE and DER++ – separately for **pretrained** backbones (model: ViT-B/16, datasets: CIFAR-100, CUB-200, ImageNet-R) and backbones trained **from scratch** (model: ResNet-18, datasets: CIFAR-100, TinyImageNet). Tables 3 and 4 report Class-IL accuracy and forgetting for the pretrained and from scratch cases respectively. Task-IL results are provided in Appendix B.

The strongest result is consistent performance on the three pretrained benchmarks. In Table 3, our full method has the highest mean accuracy and lowest mean forgetting among the conventional baselines in every setting. Its Class-IL accuracy exceeds the strongest such baseline by 12.15 percentage points on CIFAR-100, 4.79 on CUB-200, and 3.22 on ImageNet-R. These comparisons include the additional historical snapshot storage described above; they are not equal-total-memory comparisons.

Table 3: Class-IL accuracy and forgetting against continual learning baselines on three **pretrained** ViT-B/16 settings. Results are mean ± std over three seeds. Higher accuracy (↑) and lower forgetting (↓) are better.

	CIFAR-100		CUB-200		ImageNet-R	
Method	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**
SAGA (Jacobian + Replay + History)	$\mathbf{76.01 \pm 0.46}$	$\mathbf{24.24 \pm 0.43}$	$\mathbf{73.68 \pm 0.79}$	$\mathbf{15.38 \pm 0.92}$	$\mathbf{61.58 \pm 0.51}$	$\mathbf{34.48 \pm 0.40}$
DER++	57.18 ± 2.52	46.04 ± 2.88	68.89 ± 3.17	19.57 ± 3.60	58.36 ± 2.09	38.42 ± 2.36
ER-ACE	63.86 ± 3.34	38.01 ± 3.78	56.93 ± 1.79	29.34 ± 2.04	57.09 ± 1.50	38.79 ± 1.62
ER	45.61 ± 2.54	58.93 ± 2.86	64.22 ± 3.17	26.10 ± 3.43	52.85 ± 1.22	44.40 ± 1.33
SGD	9.78 ± 0.04	93.77 ± 0.15	9.02 ± 0.48	77.48 ± 0.89	9.31 ± 0.23	81.94 ± 0.18

The from scratch results define the limits of this advantage (Table 4). Our method improves on ER in accuracy on both datasets, but DER++ or ER-ACE attains higher final accuracy. On TinyImageNet it has the lowest mean forgetting, while on CIFAR-100 its forgetting is similar to DER++ and ER-ACE. The contrast is consistent with anchoring being particularly useful when adapting pretrained representations, but architecture, data, and initialization differ between the regimes, therefore these experiments do not isolate pretraining as the causal factor.

Table 4: Class-IL accuracy and forgetting against continual learning baselines with a ResNet-18 trained **from scratch**. Results are mean ± std over three seeds. Higher accuracy (↑) and lower forgetting (↓) are better.

	CIFAR-100		TinyImageNet	
Method	**Acc.**	**Forget.**	**Acc.**	**Forget.**
SAGA (Jacobian + Replay + History)	28.53 ± 0.52	38.46 ± 2.64	28.27 ± 4.43	$\mathbf{12.84 \pm 2.24}$
DER++	$\mathbf{36.28 \pm 0.24}$	37.71 ± 1.20	29.90 ± 5.00	37.61 ± 3.96
ER-ACE	32.16 ± 0.99	$\mathbf{37.41 \pm 3.88}$	$\mathbf{32.37 \pm 0.49}$	29.98 ± 2.04
ER	18.28 ± 0.39	71.89 ± 0.64	25.13 ± 0.49	52.09 ± 0.34
SGD	8.37 ± 0.71	68.55 ± 3.87	6.84 ± 0.72	53.83 ± 6.67

5.3 Effect of anchor rank r.

Table 5 studies the effect of the retained Jacobian rank $r \in 4, 8, 16, 32, 64$ for SAGA (`Jacobian + Replay + History`), using a TopK SAE with $K = 64$ and a pretrained ViT-B/16 backbone and three datasets. With raw weights, changing r also changes the effective penalty strength because the loss averages over retained directions with decaying singular values. We therefore normalize each anchor's weights to unit mean square, preserving their relative weighting while keeping the nominal penalty strength constant across ranks.

Class-IL performance generally improves, however, substantially lower ranks remain competitive: even $r \ll K$ preserves strong performance, while Task-IL accuracy is comparatively insensitive to the retained rank. This indicates that the SAE-induced geometry admits substantial low-rank compression, although retaining a larger fraction of the Jacobian-sensitive subspace can be beneficial for Class-IL performance.

Table 5: Effect of the retained Jacobian rank r for a TopK SAE with $K = 64$ on accuracy, with normalized anchor weights (constant nominal penalty strength across r) and pretrained ViT-B/16. Results are mean ± std over three seeds. Higher results are better (↑).

	CIFAR-100		ImageNet-R		CUB-200	
Configuration	**Class-IL**	**Task-IL**	**Class-IL**	**Task-IL**	**Class-IL**	**Task-IL**
Jacobian ($r = 4$) + Replay + History	75.55 ± 0.56	97.45 ± 0.09	59.41 ± 0.35	88.44 ± 0.36	70.25 ± 1.35	85.76 ± 0.52
Jacobian ($r = 8$) + Replay + History	75.71 ± 0.33	97.43 ± 0.17	60.63 ± 0.76	88.55 ± 0.48	71.87 ± 0.49	85.92 ± 0.67
Jacobian ($r = 16$) + Replay + History	76.48 ± 0.28	97.41 ± 0.04	61.64 ± 0.63	88.45 ± 0.42	74.06 ± 0.82	86.23 ± 0.34
Jacobian ($r = 32$) + Replay + History	78.08 ± 0.36	97.50 ± 0.02	63.94 ± 0.16	88.87 ± 0.24	74.52 ± 0.61	86.30 ± 0.27
Jacobian ($r = 64$) + Replay + History	79.45 ± 0.11	97.43 ± 0.13	64.34 ± 0.56	89.02 ± 0.10	74.77 ± 0.34	86.01 ± 0.74

5.4 Comparison of Low-Rank Reference Geometries

To disentangle the contribution of the SAE-induced geometry from low-rank anchoring more broadly, we compare SAE-Jacobian anchoring with random, PCA, and classifier SVD reference *geometries* of matched nominal rank for pretrained Class-IL settings (Table 6) and a from scratch setting (Table 7). All rows use normalized anchor weighting, which fixes the total trace of the quadratic metric while retaining each geometry's protected directions and relative within-subspace spectral weights (Appendix D.3). Thus PCA and classifier SVD comparisons vary more than the subspace alone, whereas the SAE geometry is additionally input dependent.

The normalized controls show that the choice of geometry matters, but not in a uniform way. SAE-Jacobian has the highest mean accuracy and lowest mean forgetting on CUB-200. On ImageNet-R, it is close to PCA and random anchoring: the 0.07-point accuracy difference from PCA is small relative to the reported standard deviations. On the other hand, classifier SVD performs best on CIFAR-100. Therefore, these results support local SAE geometry as a useful option, not a universal replacement for global subspace constructions.

Table 6: Class-IL accuracy and forgetting for different reference geometries on **pretrained** ViT-B/16. All variants use normalized anchor weighting. Results are mean ± std over three seeds. Higher accuracy (↑) and lower forgetting (↓) are better.

	CUB-200 ($r = 16$)		**ImageNet-R** ($r = 16$)		**CIFAR-100** ($r = 8$)	
Reference	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Random	70.68 ± 0.52	19.05 ± 0.84	61.11 ± 0.80	34.78 ± 1.03	71.69 ± 1.07	29.11 ± 1.12
PCA	71.64 ± 0.92	17.71 ± 1.12	61.37 ± 0.25	34.95 ± 0.30	74.98 ± 0.55	25.45 ± 0.62
Classifier SVD	67.27 ± 1.54	22.66 ± 1.67	56.69 ± 0.25	39.81 ± 0.52	$\mathbf{77.34 \pm 1.01}$	$\mathbf{22.73 \pm 0.97}$
SAE Jacobian	$\mathbf{72.55 \pm 1.19}$	$\mathbf{16.64 \pm 1.40}$	$\mathbf{61.44 \pm 0.30}$	$\mathbf{34.67 \pm 0.21}$	76.08 ± 0.64	24.16 ± 0.75

For the from scratch setting (Tab. 7), we additionally vary the nominal retained rank across all reference geometries. SAE-Jacobian is particularly effective at $r = 4$ and $r = 8$ in Class-IL accuracy, while PCA has lower mean forgetting at every displayed rank. At $r = 16$, PCA and SAE-Jacobian have nearly identical mean accuracy, and at $r = 64$ SAE-Jacobian again has higher mean accuracy. These results show that alternative geometries occupy different stability–plasticity operating points rather than supporting a universal ordering. The normalized comparison controls the total metric trace, but PCA, classifier SVD, and SAE-Jacobian retain their own relative spectral weighting; the experiment is therefore a comparison of low-rank geometries rather than of directions alone. The functional relevance of the SAE geometry is characterized by the leakage term $\beta_{s,r}$ in Eq. 12, which describes old-readout sensitivity outside the protected subspace.

Table 7: Class-IL accuracy and forgetting for different reference geometries and ranks on CIFAR-100 with ResNet-18 trained **from scratch** ($r = K$). All variants use normalized anchor weighting. Results are mean ± std over three seeds. Higher accuracy (↑) and lower forgetting (↓) are better.

	Random		**PCA**		**Classifier SVD**		**SAE Jacobian**	
Rank r	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**
4	20.83 ± 0.39	51.76 ± 2.74	19.81 ± 2.30	$\mathbf{32.57 \pm 7.77}$	21.99 ± 1.55	42.86 ± 6.33	$\mathbf{23.63 \pm 0.71}$	46.65 ± 3.06
8	20.69 ± 3.96	46.23 ± 5.20	22.48 ± 0.95	$\mathbf{36.46 \pm 7.52}$	22.52 ± 1.26	43.63 ± 8.14	$\mathbf{24.88 \pm 1.85}$	37.74 ± 2.69
16	23.39 ± 1.07	45.26 ± 2.44	$\mathbf{27.71 \pm 2.21}$	$\mathbf{27.62 \pm 7.20}$	26.29 ± 1.33	41.92 ± 1.19	27.65 ± 1.44	35.44 ± 4.94
64	23.22 ± 1.72	45.73 ± 1.71	19.07 ± 8.06	$\mathbf{28.81 \pm 6.75}$	24.39 ± 5.52	40.52 ± 5.01	$\mathbf{26.19 \pm 4.58}$	33.03 ± 4.27

6 Conclusions

We introduced Sparse Autoencoder Geometric Anchoring (SAGA), a representation-level continual learning regularizer in which task-specific sparse autoencoders induce local positive-semidefinite geometries over stored representations. Replay snapshots specify where anchoring is applied, while truncated Jacobian singular directions determine which local changes are constrained. Our results show that SAEs can serve not only as latent feature spaces, but also as sources of sample-specific geometric constraints on backbone drift. These constraints can be compressed to a chosen rank and combined with replay supervision. The strongest empirical result is the full-stack SAGA's accuracy and forgetting relative to conventional baselines across all three pretrained benchmarks.

From scratch and geometry control results show that the benefit is regime dependent, while the rank ablation shows that the SAE-induced geometry can be substantially compressed with limited performance loss. The contribution, therefore, is a concrete use of SAEs as local geometric memory, together with a theoretical analysis of what that memory controls and an empirical characterization of where it helps.

Limitations. The Jacobian is a local linear description of the SAE and, for hard TopK encoders, its exact relation to latent anchoring holds only within a fixed activation/support cell; support transitions are a genuine source of discrepancy. An unsupervised SAE also does not guarantee that its sensitive directions coincide with old-task functional sensitivity, which is why our theoretical bound contains an explicit leakage term and why the PCA/classifier/random controls are important. Results use three seeds, so small differences in means should not be over-interpreted. Historical anchors also retain complete task-boundary reservoir snapshots and their stored representations/geometric factors, so their auxiliary memory and compute grow with the number of completed tasks even though the live replay reservoir remains fixed at 500 examples. Future work should therefore investigate anchor compression, more efficient SAE/Jacobian construction, and the resulting memory–accuracy–computation trade-offs.

AI USE STATEMENT

Generative AI tools were used to assist with research ideation, mathematical checking, drafting, and document organization. The authors reviewed the AI-assisted material and independently verified the mathematical statements, experimental results, citations, implementation decisions, and final claims included in the submission. The authors take responsibility for the final content of the work. Generative AI also was used to rewrite handwritten formulas and whiteboard photos into LaTeX equations.

ETHICS STATEMENT

This work does not involve any sensitive aspects, such as human subjects, personal data, or user studies. Continual learning is an important enabler for developing adaptive machine learning systems that can incorporate new knowledge while preserving previously acquired information. Our work aims to advance this area by addressing catastrophic forgetting – its key challenge – through an anchoring method. In conducting and reporting the study, we aimed to uphold high standards of scientific excellence through transparent methodology, reproducible experimental procedures, sharing our code, making it compatible with a commonly used framework (Mammoth) and careful reporting of both strengths and limitations. We are not aware of direct harmful applications specific to the proposed method. Nevertheless, as with other machine learning methods, its use in deployed systems may inherit risks associated with the underlying application domains, and such downstream effects should be considered.

REPRODUCIBILITY STATEMENT

To support reproducibility, we evaluated our method on publicly available continual learning datasets and implement it within the widely used Mammoth continual learning framework. Our implementation follows the experimental settings and evaluation protocols described in the paper, and we provide the source code required to reproduce the experiments. The code will be made publicly available on GitHub upon acceptance.

REFERENCES

Leonard Bereska and Efstratios Gavves. Mechanistic Interpretability for AI Safety – A Review. *arXiv preprint arXiv:2404.14082*, 2024.

Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-Incremental Continual Learning into the eXtended DER-verse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark Experience for General Continual Learning: a Strong, Simple Baseline. *Advances in Neural Information Processing Systems*, 33:15920–15930, 2020.

Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New Insights on Reducing Abrupt Representation Change in Online Continual Learning. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2104.05025.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. On Tiny Episodic Memories in Continual Learning. *arXiv preprint arXiv:1902.10486*, 2019. URL https://arxiv.org/abs/1902.10486.

MohammadReza Davari, Nader Asadi, Sudhir Mudur, Rahaf Aljundi, and Eugene Belilovsky. Probing Representation Forgetting in Supervised and Unsupervised Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16712–16721, 2022.

Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. PODNet: Pooled Outputs Distillation for Small-Tasks Incremental Learning. In *European Conference on Computer Vision*, 2020. URL https://www.ecva.net/papers/eccv_2020/papers_ECCV/html/3384_ECCV_2020_paper.php.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal Gradient Descent for Continual Learning. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773, 2020. URL https://proceedings.mlr.press/v108/farajtabar20a.html.

Lucy Farnik, Tim Lawson, Conor Houghton, and Laurence Aitchison. Jacobian Sparse Autoencoders: Sparsify Computations, Not Just Activations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 16178–16236, 2025. URL https://proceedings.mlr.press/v267/farnik25a.html.

Katarzyna Filus and Sebastian Pokuciński. Measuring Monosemanticity in Sparse Autoencoders via Latent Activation Coherence. *arXiv preprint arXiv:2607.17770*, 2026. URL https://arxiv.org/abs/2607.17770.

Katarzyna Filus, Kamil Faber, Roberto Corizzo, and Christopher Kanan. Lost or hidden? a concept-level forgetting in supervised continual learning. *arXiv preprint arXiv:2605.16374*, 2026.

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.

Qiao Gu, Dongsub Shim, and Florian Shkurti. Preserving linear separability in continual learning by backward feature projection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24286–24295, 2023.

Reinhard Heckel. Provable Continual Learning via Sketched Jacobian Approximations. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 10448–10470, 2022. URL https://proceedings.mlr.press/v151/heckel22a.html.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a Unified Classifier Incrementally via Rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Hou_Learning_a_Unified_Classifier_Incrementally_via_Rebalancing_CVPR_2019_paper.html.

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*, 2024.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Haoran Li, Aditya Krishnan, Jingfeng Wu, Soheil Kolouri, Praveen K. Pilly, and Vladimir Braverman. Lifelong Learning with Sketched Structural Regularization. In *Proceedings of the 13th Asian Conference on Machine Learning*, volume 157 of *Proceedings of Machine Learning Research*, pp. 985–1000, 2021. URL https://proceedings.mlr.press/v157/li21b.html.

Zhizhong Li and Derek Hoiem. Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. doi: 10.1109/TPAMI.2017.2773081.

Alireza Makhzani and Brendan Frey. k-Sparse Autoencoders. *arXiv preprint arXiv:1312.5663*, 2013. URL https://arxiv.org/abs/1312.5663.

Sergi Masip, Gido M van de Ven, Javier Ferrando, and Tinne Tuytelaars. Putting a Face to Forgetting: Continual Learning meets Mechanistic Interpretability. *arXiv preprint arXiv:2601.22012*, 2026.

Evan Ning, Wei Xue, Dong Lou, and Yike Guo. From weights to features: Sae-guided activation regularization for llm continual learning. *arXiv preprint arXiv:2606.26629*, 2026.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual Lifelong Learning with Neural Networks: A Review. *Neural Networks*, 113:54–71, 2019.

Salah Rifai, Yann N. Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The Manifold Tangent Classifier. In *Advances in Neural Information Processing Systems*, volume 24, 2011. URL https://papers.nips.cc/paper_files/paper/2011/hash/d1f44e2f09dc172978a4d3151d11d63e-Abstract.html.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience Replay for Continual Learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021.

Eli Verwimp, Rahaf Aljundi, Shai Ben-David, Matthias Bethge, Andrea Cossu, Alexander Gepperth, et al. Continual Learning: Applications and the Road Forward. *Transactions on Machine Learning Research*, 2024.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A Comprehensive Survey of Continual Learning: Theory, Method and Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, 2024.

Honoka Yamashita, Takio Kurita, and Masaki Onishi. Continual learning by regularization in row space of weight matrix of previous task for deep neural network. In *International Conference on Agents and Artificial Intelligence*, volume 2, pp. 1138–1148, 2026.

Mingxu Zhang, Yuhan Li, Lujundong Li, Dazhong Shen, Hui Xiong, and Ying Sun. Sae-fd: Sparse autoencoder feature distillation for continual learning of large language models. *arXiv preprint arXiv:2605.25525*, 2026.

A Mathematical details of task-anchored SAE geometry

A.1 Local pullback approximation

For a frozen task-boundary SAE encoder E_s, define the nonlinear displacement energy at an anchor h_0 by

$$d_s(h_0, h_0 + \delta)^2 = \|E_s(h_0 + \delta) - E_s(h_0)\|_2^2. \tag{13}$$

Assume that E_s is differentiable near h_0 and that its Jacobian is L_s-Lipschitz in operator norm. Writing $J = J_{E_s}(h_0)$, Taylor's theorem gives

$$E_s(h_0 + \delta) - E_s(h_0) = J\delta + r(\delta), \qquad \|r(\delta)\|_2 \le \frac{L_s}{2}\|\delta\|_2^2. \tag{14}$$

Therefore

$$\left|d_s(h_0, h_0 + \delta)^2 - \delta^\top J^\top J\delta\right| \le L_s\|J\|_{\mathrm{op}}\|\delta\|_2^3 + \frac{L_s^2}{4}\|\delta\|_2^4. \tag{15}$$

Thus $J^\top J$ is the local pullback semimetric induced by Euclidean distance in SAE latent space.

A.2 Exact affine ReLU–TopK geometry

Assume

$$E(h) = \mathrm{TopK}(\mathrm{ReLU}(Wh + b)) \tag{16}$$

where TopK is the hard coordinate-masking operator used in the implementation: it keeps the K largest post-ReLU values in their original latent coordinates and sets all remaining coordinates to zero. Let h_0 be a regular point whose selected TopK coordinates are strictly positive and separated from the remaining coordinates. Let $S(h_0)$ be the selected support and let $M_{S(h_0)}$ denote the corresponding diagonal mask. Then

$$J_E(h_0) = M_{S(h_0)}W. \tag{17}$$

Consequently, for every displacement δ,

$$\delta^\top J_E(h_0)^\top J_E(h_0)\delta = \|M_{S(h_0)}W\delta\|_2^2, \tag{18}$$

so full frozen-Jacobian anchoring is exactly matching the old active preactivation changes under the frozen anchor mask.

If the ReLU/TopK mask remains unchanged on the segment $\{h_0 + \tau\delta : 0 \le \tau \le 1\}$, the encoder is affine on that segment and hence

$$E(h_0 + \delta) - E(h_0) = J_E(h_0)\delta. \tag{19}$$

Therefore the *unnormalized geometric energy* of the full Jacobian equals the SAE-latent displacement exactly inside one cell. Because $\mathrm{rank}(J_E(h_0)) \le K$, retaining every nonzero singular direction also gives equality of these per-example energies. The training reductions are not identical: the implementation divides the Jacobian energy by the retained rank r through its mean over directions, whereas the latent loss sums latent-coordinate errors. Hence fixed-cell equality should not be read as equality of the two implemented losses at the same numerical anchor weight.

A simple sufficient support-stability certificate can be written in terms of the TopK gap. Let

$$z(h) = \mathrm{ReLU}(Wh + b) \tag{20}$$

and let $z_{(1)}(h_0) \ge \cdots \ge z_{(m)}(h_0)$ be the ordered coordinates. For $K < m$, define

$$\gamma_K(h_0) = z_{(K)}(h_0) - z_{(K+1)}(h_0). \tag{21}$$

If $\gamma_K(h_0) > 0$ and

$$\|W\delta\|_\infty < \frac{\gamma_K(h_0)}{2}, \tag{22}$$

then the TopK support is unchanged. Since coordinatewise ReLU is 1-Lipschitz, every post-ReLU coordinate moves by less than $\gamma_K/2$, so the ordering across the $K/(K+1)$ gap cannot reverse. A sufficient Euclidean condition is

$$\|\delta\|_2 < \frac{\gamma_K(h_0)}{2\|W\|_{2\to\infty}}, \qquad \|W\|_{2\to\infty} = \max_j \|W_{j,:}\|_2. \tag{23}$$

Support changes can nevertheless separate latent and frozen-Jacobian anchoring sharply. For $K = 1$, $W = I_2$, $b = 0$, $a > 0$,

$$h_0 = (a + \varepsilon, a), \qquad h_1 = (a, a + \varepsilon), \tag{24}$$

we have $\|h_1 - h_0\|_2 = \sqrt{2}\,\varepsilon \to 0$, while

$$\|E(h_1) - E(h_0)\|_2^2 = 2(a + \varepsilon)^2 \to 2a^2, \tag{25}$$

but the frozen full-Jacobian penalty at h_0 is only $\varepsilon^2 \to 0$. Thus hard TopK support transitions are one mechanism by which latent matching and frozen local geometry genuinely differ even without rank truncation.

A.3 Optimal rank truncation

Let

$$Q = J^\top J = \sum_{j=1}^{q} \sigma_j^2 v_j v_j^\top, \qquad \sigma_1 \geq \cdots \geq \sigma_q > 0, \tag{26}$$

For $1 \leq r \leq q$, set $\sigma_{q+1} = 0$ and define

$$Q_r = \sum_{j=1}^{r} \sigma_j^2 v_j v_j^\top. \tag{27}$$

By the Eckart–Young–Mirsky theorem, Q_r is a best rank-r approximation to Q in both operator and Frobenius norms, with

$$\|Q - Q_r\|_{\mathrm{op}} = \sigma_{r+1}^2, \qquad \|Q - Q_r\|_F^2 = \sum_{j>r} \sigma_j^4. \tag{28}$$

For every δ,

$$0 \leq \delta^\top (Q - Q_r)\delta = \sum_{j>r} \sigma_j^2 (v_j^\top \delta)^2 \leq \sigma_{r+1}^2 \|\delta\|_2^2. \tag{29}$$

Moreover, $\mathrm{rank}(Q_r) \leq r$, so $\dim \ker Q_r \geq d - r$.

A.4 Spatial sampling and directional compression

Let μ_s^{snap} denote the distribution of input examples in the cumulative stream at task boundary s that the reservoir snapshot B_s is intended to approximate. Define the ideal nonlinear snapshot-anchor energy

$$R_s^{\mathrm{SAE}}(\theta) = \mathbb{E}_{x \sim \mu_s^{\mathrm{snap}}} \left[\|E_s(f_\theta(x)) - E_s(h_s(x))\|_2^2 \right]. \tag{30}$$

With $\delta_s(x;\theta) = f_\theta(x) - h_s(x)$, set

$$q_s(x;\theta) = \delta_s^\top Q_s(h_s(x))\delta_s, \qquad q_{s,r}(x;\theta) = \delta_s^\top Q_{s,r}(x)\delta_s, \tag{31}$$

where Q_s is the full pullback metric from Eq. 2. Then

$$\left| R_s^{\mathrm{SAE}}(\theta) - \widehat{R}_{s,r}(\theta) \right| \leq \underbrace{\mathbb{E}_{x \sim \mu_s^{\mathrm{snap}}} \left| \|E_s(h_s + \delta_s) - E_s(h_s)\|_2^2 - q_s(x;\theta) \right|}_{\mathcal{E}_{\mathrm{local}}} \tag{32}$$

$$+ \underbrace{\mathbb{E}_{x \sim \mu_s^{\mathrm{snap}}} \left[q_s(x;\theta) - q_{s,r}(x;\theta) \right]}_{\mathcal{E}_{\mathrm{rank}}} \tag{33}$$

$$+ \underbrace{\left| \mathbb{E}_{x \sim \mu_s^{\mathrm{snap}}} q_{s,r}(x;\theta) - \frac{1}{|B_s|} \sum_{x_i \in B_s} q_{s,r}(x_i;\theta) \right|}_{\mathcal{E}_{\mathrm{sample}}}. \tag{34}$$

The three terms quantify local/support error, directional compression, and finite spatial sampling. The rank term obeys

$$\mathcal{E}_{\mathrm{rank}} \leq \mathbb{E}_{x \sim \mu_s^{\mathrm{snap}}} \left[\sigma_{s,r+1}(x)^2 \|\delta_s(x;\theta)\|_2^2 \right], \tag{35}$$

and $\mathcal{E}_{\mathrm{local}} = 0$ inside a fixed affine ReLU–TopK cell. This decomposition is written for the unnormalized geometric energy $\widehat{R}_{s,r}$; the raw Jacobian training loss carries the additional factor $1/r$ from Eq. 6.

For intuition about finite spatial resolution, fix the entire loss function, including h_s, E_s, r, and θ, independently of a hypothetical i.i.d. evaluation sample B_s from μ_s^{snap}. If $0 \leq q_{s,r}(x;\theta) \leq M_s$ almost surely, Hoeffding's inequality gives, with probability at least $1 - \eta$,

$$\mathcal{E}_{\mathrm{sample}} \leq M_s \sqrt{\frac{\log(2/\eta)}{2|B_s|}}. \tag{36}$$

This is an illustrative fixed-model sampling statement, not a generalization guarantee for the trained continual learner: the implementation uses reservoir snapshots to fit the SAE and later optimize the backbone, so the independence assumption does not hold for those same stored examples.

A.5 FUNCTIONAL LEAKAGE OUTSIDE THE PROTECTED SUBSPACE

Let $g_s : \mathbb{R}^d \to \mathbb{R}^{c_s}$ be a frozen old-task logit map and let $A_s = J_{g_s}(h_0)$. Let $Q_r = V_r \Sigma_r^2 V_r^\top$ be formed from r retained *nonzero* singular values, with $\sigma_r > 0$, and set $P_r = V_r V_r^\top$. Define

$$\alpha_{s,r} = \|A_s V_r \Sigma_r^{-1}\|_{\mathrm{op}}, \qquad \beta_{s,r} = \|A_s(I - P_r)\|_{\mathrm{op}}. \tag{37}$$

If g_s has an L_g-Lipschitz Jacobian on a convex neighborhood containing the displacement segment, Taylor expansion gives

$$g_s(h_0 + \delta) - g_s(h_0) = A_s\delta + \rho(\delta), \qquad \|\rho(\delta)\|_2 \le \frac{L_g}{2}\|\delta\|_2^2. \tag{38}$$

Decomposing $A_s\delta = A_s P_r \delta + A_s(I - P_r)\delta$ and using

$$A_s P_r \delta = A_s V_r \Sigma_r^{-1}(\Sigma_r V_r^\top \delta) \tag{39}$$

yields

$$\|g_s(h_0 + \delta) - g_s(h_0)\|_2 \le \alpha_{s,r}\sqrt{\delta^\top Q_r \delta} + \beta_{s,r}\|\delta\|_2 + \frac{L_g}{2}\|\delta\|_2^2. \tag{40}$$

Thus a low-rank SAE anchor can control old-task function only to the extent that old-task sensitivity outside the protected subspace, described by $\beta_{s,r}$, is small.

This statement concerns compatibility with the frozen old-task readout. In the Class-IL experiments the live classifier continues to change, so classifier-parameter drift and competition from newly introduced classes can alter logits even when the representation remains compatible with the old frozen readout.

For a linear old readout $g_s(h) = A_s h + b_s$ and any $Q \succeq 0$, there exists a finite c such that

$$\|A_s\delta\|_2^2 \le c\,\delta^\top Q \delta \qquad \text{for all } \delta \tag{41}$$

if and only if

$$\ker Q \subseteq \ker A_s. \tag{42}$$

When this holds, the smallest constant is

$$c_\star = \|A_s Q^{\dagger/2}\|_{\mathrm{op}}^2. \tag{43}$$

This gives an exact algebraic version of the same requirement: no old-task-sensitive direction may lie completely in the anchor nullspace.

A.6 SAME-POINT HISTORICAL QUADRATIC AGGREGATION

The following identity applies when several historical quadratic anchors constrain the *same* current representation h. It is useful for interpreting how multiple local PSD constraints combine, but it is not a literal reduction of the full implementation, where different task-boundary snapshots generally contain different replay examples and are evaluated on independently sampled minibatches. Let

$$L_{\mathrm{hist}}(h) = \sum_{s=1}^{T} \omega_s (h - h_s)^\top Q_s (h - h_s), \qquad \omega_s > 0, \tag{44}$$

with $Q_s \succeq 0$. Define

$$Q_H = \sum_s \omega_s Q_s, \qquad b_H = \sum_s \omega_s Q_s h_s, \qquad c_H = \sum_s \omega_s h_s^\top Q_s h_s. \tag{45}$$

With $h_\star = Q_H^\dagger b_H$,

$$L_{\mathrm{hist}}(h) = (h - h_\star)^\top Q_H (h - h_\star) + \kappa, \tag{46}$$

where $\kappa = c_H - b_H^\top Q_H^\dagger b_H \ge 0$. Since all Q_s are positive semidefinite,

$$\ker Q_H = \bigcap_{s=1}^{T} \ker Q_s. \tag{47}$$

Thus, at a common point, adding PSD historical anchors can only shrink the common unconstrained subspace. In the actual training procedure, History instead averages or sums stochastic losses from separately stored task-boundary snapshots; the default is the mean reduction, so total nominal anchor-loss scale does not grow linearly with the number of snapshots, although storage and compute do.

B TASK-IL RESULTS

For completeness, we report Task-IL accuracy and forgetting corresponding to the main Class-IL experiments. Task-IL assumes that the task identity is available at test time and serves as a secondary diagnostic.

Table 8: Task-IL accuracy and forgetting for SAE anchoring ablations with a pretrained ViT-B/16. Results are mean ± std over three seeds.

	CIFAR-100		CUB-200		ImageNet-R	
Configuration	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Vanilla Latent	89.34 ± 0.30	9.07 ± 0.42	74.97 ± 0.29	10.30 ± 0.32	84.29 ± 0.62	8.65 ± 0.45
Vanilla Jacobian	93.10 ± 0.59	5.96 ± 0.67	79.23 ± 1.49	9.62 ± 1.38	84.00 ± 0.53	9.83 ± 0.65
Latent + History	95.66 ± 0.12	2.79 ± 0.15	82.28 ± 0.81	4.61 ± 0.83	87.33 ± 0.60	6.41 ± 0.45
Jacobian + History	96.57 ± 0.13	2.00 ± 0.12	84.21 ± 0.22	4.49 ± 0.26	87.30 ± 0.22	6.15 ± 0.53
Latent + Replay	94.59 ± 0.37	3.24 ± 0.37	81.01 ± 0.43	4.35 ± 0.53	86.66 ± 0.28	6.27 ± 0.14
Jacobian + Replay	97.17 ± 0.03	1.53 ± 0.06	84.56 ± 0.75	4.08 ± 0.68	87.24 ± 0.20	6.37 ± 0.32
Latent + Replay + History	97.24 ± 0.12	1.10 ± 0.17	84.85 ± 0.78	$\mathbf{2.30 \pm 0.59}$	$\mathbf{89.00 \pm 0.25}$	$\mathbf{4.66 \pm 0.02}$
SAGA (Jacobian + Replay + History)	$\mathbf{97.54 \pm 0.10}$	$\mathbf{1.03 \pm 0.18}$	$\mathbf{86.38 \pm 0.34}$	2.42 ± 0.47	88.76 ± 0.39	$\mathbf{4.66 \pm 0.23}$

Table 9: Task-IL accuracy and forgetting for SAE anchoring ablations with a ResNet-18 trained from scratch.

	CIFAR-100		TinyImageNet	
Configuration	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Vanilla Latent	59.03 ± 4.49	9.12 ± 2.16	60.86 ± 1.28	1.78 ± 0.24
Vanilla Jacobian	54.51 ± 8.59	21.82 ± 3.10	47.29 ± 5.19	22.41 ± 2.19
Latent + History	65.88 ± 2.20	2.89 ± 0.63	61.34 ± 0.66	2.01 ± 0.19
Jacobian + History	68.50 ± 5.34	5.79 ± 0.92	59.19 ± 3.94	3.47 ± 0.44
Latent + Replay	67.19 ± 1.47	9.21 ± 2.46	56.24 ± 2.06	7.94 ± 1.43
Jacobian + Replay	68.50 ± 1.64	11.25 ± 2.02	51.98 ± 6.02	16.55 ± 4.48
Latent + Replay + History	72.16 ± 0.66	3.97 ± 0.66	58.61 ± 1.46	$\mathbf{2.31 \pm 0.41}$
SAGA (Jacobian + Replay + History)	$\mathbf{74.35 \pm 0.43}$	$\mathbf{3.93 \pm 0.90}$	$\mathbf{61.84 \pm 3.38}$	3.04 ± 1.34

Overall, the Task-IL results are consistent with the Class-IL findings included in the main paper. In particular, combining replay with persistent task history remains the strongest SAE-anchoring configuration, while SAE-Jacobian anchoring shows particularly strong performance in the pretrained setting, as well compared to the SAE latent anchoring in the from scratch scenario. Our method also consistently reduces forgetting relative to standard baselines.

Table 10: Task-IL accuracy and forgetting against continual learning baselines on pretrained ViT-B/16.

	CIFAR-100		CUB-200		ImageNet-R	
Method	**Acc.**	**Forget.**	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Ours	$\mathbf{97.54 \pm 0.10}$	$\mathbf{1.03 \pm 0.18}$	$\mathbf{86.38 \pm 0.34}$	$\mathbf{2.42 \pm 0.47}$	$\mathbf{88.76 \pm 0.39}$	$\mathbf{4.66 \pm 0.23}$
DER++	91.64 ± 0.91	7.90 ± 0.87	83.14 ± 0.98	5.30 ± 1.03	85.77 ± 0.49	8.42 ± 0.53
ER-ACE	93.90 ± 1.10	5.11 ± 1.24	77.92 ± 1.46	7.83 ± 1.65	84.10 ± 0.38	9.97 ± 0.40
ER	95.16 ± 0.33	3.94 ± 0.39	83.01 ± 0.75	5.76 ± 0.79	86.00 ± 0.58	7.82 ± 0.68
SGD	74.31 ± 1.13	22.07 ± 1.37	39.48 ± 1.03	43.63 ± 1.21	53.57 ± 1.96	32.78 ± 2.16

Table 11: Task-IL accuracy and forgetting against continual learning baselines with a ResNet-18 trained from scratch.

	CIFAR-100		TinyImageNet	
Method	**Acc.**	**Forget.**	**Acc.**	**Forget.**
Ours	74.35 ± 0.43	$\mathbf{3.93 \pm 0.90}$	61.84 ± 3.38	$\mathbf{3.04 \pm 1.34}$
DER++	$\mathbf{75.15 \pm 0.24}$	8.03 ± 1.14	66.08 ± 4.69	8.81 ± 3.47
ER-ACE	70.00 ± 0.56	12.34 ± 3.40	61.10 ± 0.55	12.88 ± 1.09
ER	68.13 ± 0.86	17.28 ± 0.71	$\mathbf{66.09 \pm 0.31}$	9.65 ± 0.44
SGD	32.49 ± 3.01	41.89 ± 2.35	17.13 ± 2.25	42.96 ± 5.99

C Effect of SAE dictionary size (overcompleteness)

We perform all of our main experiments with SAE dictionary of size 256, i.e. an *undercomplete* dictionary relative to the ResNet-18 and ViT feature dimensions ($D = 512$, an expansion factor of $0.5\times$), whereas SAEs are more commonly trained overcomplete when used for representational disentanglement and interpretability. Table 12 tests different dictionary sizes on CIFAR-100 with ResNet-18, ranging from $0.5\times$ (the default for dictionary size 256 in the ResNet-18 case) up to $8\times$, at fixed `sae_topk` $=$ `jacobian_k` $= 16$. Class-IL accuracy is rather flat across dictionary sizes and within one standard deviation of the default 256-unit dictionary at every size tested. The only configuration that is both worse and markedly less stable across seeds is dictionary size 1024 (expansion factor 2). Our obtained results suggest that overcomplete dictionaries are not essential for the proposed anchoring mechanism, however, they can still be used when interpretability of continual learning models representations is an additional objective, while the same SAE can be used to derive the Jacobian-based anchors for continual learning.

Table 12: Effect of SAE dictionary size on accuracy for CIFAR-100 with a ResNet-18 trained from scratch, at fixed $K = r = 16$. Expansion factor is relative to the backbone's 512-dimensional feature space. Results are mean $\pm$ std over three seeds. Higher results are better (↑).

Dictionary size	**Expansion**	**Class-IL**	**Task-IL**
256	$0.5\times$	$\mathbf{28.53 \pm 0.52}$	$\mathbf{74.35 \pm 0.43}$
512	$1\times$	26.67 ± 0.87	71.29 ± 0.63
1024	$2\times$	24.66 ± 5.33	69.49 ± 4.39
2048	$4\times$	28.07 ± 1.41	71.80 ± 1.09
4096	$8\times$	28.10 ± 2.15	71.51 ± 1.22

D Detailed experimental setup

D.1 Datasets

Table 13 summarizes the datasets. Classes are assigned to tasks in the class order of the dataset (no class permutation), so the task sequence is identical for every seed; each task is trained for a fixed number of epochs (Table 14). CIFAR-100 is used at native 32×32 resolution with a from scratch ResNet-18 and upsampled to 224×224 with the pretrained ViT-B/16. TinyImageNet has 500 training images per class and its official validation split serves as the test set. CUB-200 (fine-grained bird species) has about 30 training images per class and ImageNet-R (renditions such as art and cartoons of 200 ImageNet classes) 120 per class, with the fixed train/test split of the respective data release. In all settings the seed changes the initialization of the new layers and of the SAE, the data order and the buffer contents, but not the task sequence.

Datasets in training from scratch A network trained from scratch needs many images per class. CIFAR-100 and TinyImageNet provide 500 training images per class, whereas ImageNet-R provides 120 (24,000 images for 200 classes, moreover renditions of varying style) and CUB-200 only 30 (5,994 images). In preliminary runs of a ResNet-18 trained from scratch on these two datasets (10 epochs per task, three seeds), all methods, including plain SGD, stayed below 10% Class-IL accuracy, so they cannot be meaningfully compared. In contrast, a ResNet-18 trained from scratch

Table 13: Datasets. "Tasks" is the number of tasks × classes per task. Input: image size seen by the network. Train-time augmentation is listed in the text; test images are only resized (and, where stated, center-cropped) and normalized.

Dataset	Classes	Tasks	Train / Test images	Input	Backbone
CIFAR-100	100	10×10	50,000 / 10,000	32×32	ResNet-18 (scratch)
TinyImageNet	200	10×20	100,000 / 10,000	64×64	ResNet-18 (scratch)
CIFAR-100	100	10×10	50,000 / 10,000	224×224	ViT-B/16 (pretrained)
ImageNet-R	200	10×20	24,000 / 6,000	224×224	ViT-B/16 (pretrained)
CUB-200	200	10×20	5,994 / 5,794	224×224	ViT-B/16 (pretrained)

on TinyImageNet reaches 25–32% with the rehearsal-based methods (and for SGD: 6.8%). We therefore evaluate the from-scratch regime on CIFAR-100 and TinyImageNet and the pretrained regime on the 224-pixel datasets (upscaled CIFAR-100, ImageNet-R and CUB-200).

D.2 Hyperparameters

Table 14 lists the tuned hyperparameters of our method (learning rate and anchor weight) together with the fixed training settings, for the main results. The SAE-specific hyperparameters are fixed in all of them (`sae_epochs` $= 20$; `sae_lr` $= 10^{-3}$; dictionary size 256).

Table 14: Tuned hyperparameters and fixed training settings for our method SAGA (Jacobian + Replay + History), by setting.

Setting	Backbone	lr	Anchor weight	Epochs/task	Buffer	Optimizer	Batch
CIFAR-100	ViT-B/16 (pretrained)	3×10^{-5}	3.0	10	500	Adam	64
ImageNet-R	ViT-B/16 (pretrained)	3×10^{-5}	0.3	10	500	Adam	64
CUB-200	ViT-B/16 (pretrained)	3×10^{-5}	0.3	10	500	Adam	64
CIFAR-100	ResNet-18 (scratch)	0.1	0.3	20	500	SGD	64
TinyImageNet	ResNet-18 (scratch)	0.03	1.0	20	5120	SGD	64

For every setting, our method's learning rate and anchor weight were selected by a grid search (learning rate × anchor weight) evaluated on a held-out validation split (10% of the training set held out per class), using a single screening seed. All numbers reported elsewhere in the paper are then produced by a separate run at the winning hyperparameters, over three seeds on the separate test set.

D.3 Comparisons with reference geometries

To test whether SAE-Jacobian anchoring benefits from SAE-induced geometry specifically or from low-rank anchoring more generally, we compare it with random, PCA, and classifier SVD geometries under the same Replay+History protocol. Each geometry supplies an orthonormal basis V_r and nonnegative direction weights S_r, defining

$$Q_r^{(m)} = V_r^{(m)} S_r^{(m)2} V_r^{(m)\top}. \tag{48}$$

For these comparisons we use `anchor_weight_norm=normalized`, which rescales the weights according to

$$\widetilde{S}_r = \frac{S_r}{\sqrt{\frac{1}{r}\sum_{i=1}^{r} S_i^2}}. \tag{49}$$

Because the implementation also averages squared projected drift over the r retained directions, the normalized per-example loss is

$$\frac{1}{r}\|\widetilde{S}_r V_r^\top \delta\|_2^2 = \frac{\delta^\top Q_r^{(m)} \delta}{\operatorname{tr} Q_r^{(m)}}. \tag{50}$$

Thus the comparison normalizes the total metric trace while preserving each geometry's directions and relative within-subspace spectral weighting.

The concrete reference geometries are:

- *Random*: draw r Gaussian directions and orthonormalize them by QR. The directions use unit weights, so $Q_r = V_r V_r^\top$ and the implemented loss is $\|V_r^\top \delta\|_2^2/r$. The seed of the basis is derived from the run seed and the task index.
- *PCA*: center the same representation dataset that would otherwise be used to fit the SAE, compute $H_c = USV^\top$, and use the leading right-singular directions together with their singular values. Before normalization, $Q_r^{\mathrm{PCA}} = V_r S_r^2 V_r^\top$.
- *Classifier SVD*: decompose the frozen classifier weight matrix $W = USV^\top$ and use its leading right-singular directions and singular values. Before normalization, the full-rank energy satisfies $\|SV^\top \delta\|_2^2 = \|W\delta\|_2^2$, i.e. squared drift of a frozen linear readout. Under Eq. 50, the implemented full-rank quantity is proportional to logit drift, $\|W\delta\|_2^2/\|W\|_F^2$. The requested rank is capped by the numerical rank of W.
- *SAE Jacobian*: use the per-example right-singular directions and singular values of the local SAE encoder Jacobian. Unlike the other three controls, this basis varies across anchor examples.

PCA and classifier SVD therefore vary both the protected directions and their within-subspace spectral weights. The experiment should be interpreted as a comparison of alternative low-rank geometries rather than of directions alone.

D.4 SAE SETUP

At every task boundary a fresh TopK SAE Gao et al. (2025) is trained on the frozen backbone's features of the reservoir snapshot; its encoder Jacobian is evaluated at the snapshot's samples and, in the Jacobian mode used for all main results, the SAE is then discarded, keeping only the low-rank directions and weights.

Architecture. SAEs act on the pooled pre-logit feature $h \in \mathbb{R}^D$ (class token after the final layer normalization for ViT-B/16, $D = 768$; globally average-pooled last-stage output for ResNet-18, $D = 512$), used without centering or rescaling. It has one hidden layer with a hard TopK activation,

$$E(h) = \mathrm{TopK}_K\big(\mathrm{ReLU}(W_e h + b_e)\big), \qquad \hat{h} = W_d E(h) + b_d, \tag{51}$$

with $W_e \in \mathbb{R}^{m\times D}$ and $W_d \in \mathbb{R}^{D\times m}$; TopK_K keeps the K largest post-ReLU activations of each sample in place and zeroes the rest. The weights are untied, the decoder columns are unconstrained, initialization is the PyTorch default, and there are $2Dm + m + D$ parameters. The values used for the main results are listed in Table 15: $m = 256$ and $K = 16$ (undercomplete: $0.33\times$ the feature dimension for ViT-B/16, $0.5\times$ for ResNet-18). The ablations change them as follows: $K = 64$ for all ranks $r \in \{4, 8, 16, 32, 64\}$ in the rank ablation, so that the SAE is identical across ranks; $K = r$ in the reference-geometry comparison, except $K = 16$ at $r = 4$; and $m \in \{256, \ldots, 4096\}$ at $K = r = 16$ in the dictionary-size ablation.

Training. The SAE minimizes only the mean squared reconstruction error over minibatches $\mathcal{B}$,

$$\mathcal{L}_{\mathrm{SAE}} = \frac{1}{|\mathcal{B}|\, D} \sum_{h\in\mathcal{B}} \big\|\hat{h} - h\big\|_2^2, \tag{52}$$

with no sparsity penalty (TopK provides the sparsity), auxiliary loss or weight decay. It uses Adam (learning rate 10^{-3}, default momenta) for 20 epochs with batch size 64, reshuffled every epoch, independently of the continual learner's optimizer. It is initialized from scratch at every boundary and then frozen in evaluation mode: 8 steps per epoch (160 in total) for 500 samples, 1600 steps for the 5120 samples of TinyImageNet. These values are identical in all settings and were not tuned.

Training data. The SAE is trained without labels on the features of all $N = \min(\text{images seen}, \text{buffer capacity})$ images in the reservoir buffer at the boundary ($N \le 500$; $N \le 5120$ for TinyImageNet). As in the baselines, the reservoir is updated at every training step, so it approximates a uniform sample of the stream seen so far, contains all classes seen up to that task, and, when a task is trained for several epochs, may hold the same image more than once. Images are stored without augmentation and passed through the deterministic part of the dataset's test-time preprocessing (resizing, normalization), as when the anchor loss compares h_{new} with h_{old}; features come from the backbone at the boundary in evaluation mode.

From the SAE to the geometry. For a buffer image x with feature h_{old}, let $\pi_1(x), \ldots, \pi_K(x)$ index its K largest post-ReLU activations $a = \text{ReLU}(W_e h_{\text{old}} + b_e)$. Away from ReLU and TopK boundaries the encoder is locally linear and its Jacobian is read from the weights in closed form:

$$J_x = \left[\mathbb{1}[a_{\pi_i(x)} > 0] \, (W_e)_{\pi_i(x),:} \right]_{i=1}^{K} \in \mathbb{R}^{K \times D}. \tag{53}$$

We take the thin SVD $J_x = U \Sigma V^\top$ (in chunks of 64 samples), keep the $r \leq K$ leading σ_i and v_i, and store them with h_{old}, the image and its label in the snapshot; the normalized weighting rescales σ to $\tilde{\sigma}$ (Eq. 49). In Latent mode the frozen SAE is kept and $E(h_{\text{old}}) \in \mathbb{R}^m$ is stored instead. Anchors therefore need only the buffer images, the backbone at the boundary and the SAE (transiently, while the directions are computed); no data of the finished task, old-task classifier or gradients are used. The reference geometries of Section 5.4 train no SAE: *random* needs only D, r and a seed (Gaussian matrix orthonormalized by QR; seed derived from the run seed and task index), *PCA* the centered buffer features (leading right singular vectors and values), *classifier SVD* the classifier weights at the boundary. These bases are global, one $D \times r$ matrix shared by all anchors, whereas SAE-Jacobian directions are per anchor.

Table 15: SAE hyperparameters for the main results. D: feature dimension; m: dictionary size; K: TopK sparsity; r: retained Jacobian rank; N: buffer samples per snapshot (SAE training set and anchors); SAE parameters: $2Dm + m + D$. The training hyperparameters (Adam, learning rate 10^{-3}, 20 epochs, batch size 64) are identical in all settings.

Backbone; datasets	D	m	K	r	N	**SAE params**
ViT-B/16; CIFAR-100, CUB-200, ImageNet-R	768	256	16	16	500	394k
ResNet-18; CIFAR-100	512	256	16	16	500	263k
ResNet-18; TinyImageNet	512	256	16	16	5120	263k

E Computational Overhead

Table 16 reports the mean wall-clock training time per task for our method and SGD, included as the least computationally expensive baseline, across five different settings for which per-task timing was recorded. Independently measured GPU-device time using CUDA events differs from wall-clock time by less than 1% in all cases, indicating that the runs are predominantly GPU-bound with negligible CPU and data-loading overhead. We therefore omit GPU-device time as a separate column.

Table 16: Mean wall-clock time per task (seconds), our method vs. SGD, across the five settings with timing instrumentation. Ratio = Ours / SGD.

Setting	**Ours (s/task)**	**SGD (s/task)**	**Ratio**
CUB-200 (ViT-B/16, pretrained)	114.3	16.6	6.9×
ImageNet-R (ViT-B/16, pretrained)	429.8	67.0	6.4×
CIFAR-100 (ViT-B/16, pretrained)	904.2	138.6	6.5×
CIFAR-100 (ResNet-18, scratch)	82.6	12.9	6.4×
TinyImageNet (ResNet-18, scratch)	302.2	42.8	7.1×

The proposed method introduces a consistent computational overhead across all evaluated settings, with training time remaining within a narrow $6.4\times$–$7.1\times$ range relative to SGD. This indicates that the additional cost is predictable and primarily comes from SAE training, Jacobian/SVD computation at task boundaries, and the additional anchoring passes during optimization.