SAE-FD: Sparse Autoencoder Feature Distillation for Continual Learning of Large Language Models

Anonymous ACL submission

Abstract

Continual learning enables large language models to adapt to evolving tasks without retraining from scratch, yet catastrophic forgetting remains a central obstacle. Among continual learning methods, regularization-based approaches are widely used to constrain model updates and reduce forgetting, operating in weight space, gradient space, or output space. However, these dense representation spaces suffer from feature superposition, where multiple concepts are encoded in overlapping dimensions, making it difficult to selectively protect previously learned knowledge without impeding new-task learning. To address this issue, we propose SAE-FD (Sparse Autoencoder Feature Distillation), which anchors model representations in the sparse feature space of a pretrained Sparse Autoencoder, where dense activations are decomposed into a sparse overcomplete basis that reduces representational entanglement, enabling more targeted regularization with less interference to new-task learning. Experiments on two continual learning benchmarks across three model architectures show that SAE-FD consistently outperforms existing regularization-based methods, achieving up to 52.70% average accuracy with only -0.46% backward transfer.

1 Introduction

Large language models (LLMs) have demonstrated strong capabilities across diverse domains, including molecular science (Zhang et al., 2025b,a) and social science (Cui et al., 2026). In practice, deploying LLMs for specific application scenarios often requires fine-tuning on specific data. When models must be sequentially adapted to multiple tasks over time, catastrophic forgetting (Kirkpatrick et al., 2017) becomes a central challenge: training on a new task degrades performance on previously learned tasks due to the overwriting and interference of internal representations. To mitigate forgetting, existing continual learning (CL) methods regularize model updates in different representation spaces. Weight-space methods such as EWC (Kirkpatrick et al., 2017) penalize changes to important parameters; gradient-space methods such as O-LoRA (Wang et al., 2023a) and GORP (Wang et al., 2025) project updates to be orthogonal to previous task subspaces; output-space methods such as LwF (Li and Hoiem, 2017) and SEEKR (He et al., 2024) distill model outputs or attention patterns. A common limitation is that all three families operate in dense representation spaces where features representing different tasks are superimposed (Elhage et al., 2022), making it difficult to selectively protect old-task knowledge without constraining new-task learning.

Sparse Autoencoders (SAEs) (Cunningham et al., 2024; Bricken et al., 2023), developed for mechanistic interpretability, decompose dense activations into sparse, overcomplete feature vectors ($D \gg d$) that disentangle superimposed representations. The high sparsity and the overcomplete basis provide a promising decoupled space for continual learning regularization, where the reduced entanglement allows more targeted protection of previously learned knowledge. However, this application is non-trivial, as it requires training model-specific SAEs, designing loss functions that maintain gradient flow through the sparse encoding, and adapting regularization strength across different tasks and training stages.

To address these challenges, we introduce SAE-FD (Sparse Autoencoder Feature Distillation), a continual learning method that performs representation anchoring in SAE feature space. SAE-FD operates in three stages. First, we train a Gated SAE on diverse text activations from the base model to obtain a high-quality sparse feature decomposition. Second, after fine-tuning on each task using LoRA, we capture per-token SAE feature activations on a small set of anchor samples, creating a compact snapshot of the model's learned representations.

Third, during subsequent task training, we apply a sparse-aware distillation loss that combines cosine direction preservation with active-feature magnitude matching. The total training loss combines the task loss with the distillation loss weighted by a coefficient λ; since both losses vary in magnitude across tasks and training steps, a fixed λ cannot maintain a consistent regularization effect. We therefore introduce an adaptive mechanism that adjusts λ to target a fixed contribution ratio of the distillation loss, providing strong protection at the start of each new task and relaxing as the model converges. By operating in the sparse feature space, SAE-FD reduces the interference between regularization and new-task learning, achieving a better plasticity-stability tradeoff than dense-space alternatives.

In conclusion, our contributions are:

- We propose SAE-FD, the first continual learning method that performs regularization in the sparse, decoupled feature space of SAEs, enabling more targeted preservation of previously learned knowledge than baselines.
- We design an adaptive λ mechanism that dynamically adjusts regularization strength during training by targeting a fixed contribution ratio of the distillation loss to the total loss, providing stronger protection when forgetting risk is high and relaxing as the model stabilizes.
- Experiments on two CL benchmarks across three model families show that SAE-FD achieves 52.70% average accuracy with only -0.46% backward transfer on TRACE, outperforming the strongest published baseline by +2.3% AA with 34% less forgetting.

2 Related Work

Continual Learning for LLMs. Parameter-efficient fine-tuning via LoRA (Hu et al., 2022) has become the dominant LLM adaptation strategy, enabling a family of LoRA-based CL methods. O-LoRA (Wang et al., 2023a) maintains orthogonal subspaces across tasks; GORP (Wang et al., 2025) projects gradients into low-rank subspaces optimized for minimal interference; TreeLoRA (Qian et al., 2025) organizes task-specific LoRA modules in a hierarchical structure with knowledge distillation; N-LoRA (Yang et al., 2025) addresses parameter collision between tasks; InfLoRA (Liang and Li, 2024) constructs interference-free low-rank subspaces. Prompt-based methods including L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), and Progressive Prompts (Razdaibiedina et al., 2023) prepend learnable tokens to model inputs, but their effectiveness is limited on generation tasks. Replay-based methods maintain a memory buffer of previous-task exemplars; SEEKR (He et al., 2024) combines selective attention distillation with replay, achieving strong results but requiring stored training data and full fine-tuning.

Representation Distillation for CL. A complementary line of work preserves previous-task knowledge through distillation at various levels of model internals. At the output level, LwF (Li and Hoiem, 2017) distills logits from a frozen teacher, and DER++ (Buzzega et al., 2020) stores logits alongside replay exemplars. At the attention level, SEEKR (He et al., 2024) identifies and distills the most task-relevant attention patterns, and DATA (Liao et al., 2025) decomposes attention for task adaptation. At the hidden state level, GeRe (Zhang et al., 2025c) enforces activation consistency on anchor samples, demonstrating that feature-level distillation outperforms logit-level distillation, though their ablation shows that raw L1/L2 matching of dense activations is suboptimal. SAE-FD distills in the sparse, decoupled feature space of an SAE, reducing representational entanglement before applying regularization.

Sparse Autoencoders. SAEs learn to decompose neural network activations into sparse, overcomplete features (Cunningham et al., 2024; Bricken et al., 2023), and recent variants such as Gated SAEs (Rajamanoharan et al., 2024) achieve high-quality reconstruction with sparser features, scaling reliably to production-scale models (Templeton et al., 2024; Gao et al., 2024). The superposition hypothesis (Elhage et al., 2022), which posits that networks represent more features than they have dimensions by exploiting sparsity, provides theoretical grounding for why SAE-based decomposition can disentangle overlapping representations. Beyond interpretability, SAEs have been applied to controllable generation, where Zhang et al. (2026) show that SAE features can serve as decoupled intervention handles for steering molecular properties in LLMs, demonstrating the practical utility of SAE-based disentanglement. To our knowledge, no prior work has explored SAE feature space as a regularization target for continual learning.

3 Preliminaries

Continual Learning. We consider a sequence of T tasks $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$, where each task $\mathcal{D}_t = \{(x_i^t, y_i^t)\}$ consists of input-output pairs. The model is trained sequentially on each task. A central challenge in this setting is catastrophic forgetting: as the model adapts to later tasks, performance on earlier tasks may degrade due to interference between task-specific representations. To quantify this, we use two standard metrics. Let $a_{i,t}$ denote the performance on task t after training through task i:

$$\mathrm{AA}_T = \frac{1}{T} \sum_{t=1}^{T} a_{T,t} \tag{1}$$

$$\mathrm{BWT}_T = \frac{1}{T-1} \sum_{t=1}^{T-1} (a_{T,t} - a_{t,t}) \tag{2}$$

Average Accuracy (AA) measures the overall performance across all tasks after the final training stage; higher AA indicates stronger overall capability. Backward Transfer (BWT) captures how much each task's performance degrades after subsequent training; a BWT of zero indicates no forgetting, and higher (less negative) BWT is better. An ideal continual learner achieves both high AA (strong performance on all tasks) and BWT close to zero (no forgetting of previously learned tasks).

Sparse Autoencoder. Sparse Autoencoders (SAEs) (Cunningham et al., 2024) are dictionary learning models trained to decompose dense neural network activations into sparse linear combinations of learned feature directions. Given an activation vector $\mathbf{h} \in \mathbb{R}^d$, an SAE maps it to a higher-dimensional feature space $\mathbb{R}^D$ ($D \gg d$), where the resulting feature vector is sparse: only a small fraction of features are active for any given input. This overcomplete, sparse representation disentangles superimposed concepts in the original activation space, producing features that are more decoupled than the raw dimensions (Bricken et al., 2023).

In this work, we adopt the Gated SAE variant (Rajamanoharan et al., 2024), which separates the encoding process into gating and magnitude pathways:

$$\mathbf{g} = \sigma(W_{\text{gate}}(\mathbf{h} - \mathbf{b}_{\text{dec}}) + \mathbf{b}_{\text{gate}}) \tag{3}$$

$$\mathbf{m} = W_{\text{enc}}(\mathbf{h} - \mathbf{b}_{\text{dec}}) + \mathbf{b}_{\text{enc}} \tag{4}$$

$$\mathbf{f}_{\text{pre}} = \mathbf{g} \odot \mathbf{m} \tag{5}$$

$$\mathbf{f} = \mathrm{ReLU}(\mathbf{f}_{\text{pre}}) \tag{6}$$

where σ is the sigmoid function, $W_{\text{gate}}, W_{\text{enc}} \in \mathbb{R}^{D \times d}$, $\odot$ denotes element-wise multiplication, and $\mathbf{b}_{\text{dec}} \in \mathbb{R}^d$ is the decoder bias used for input centering. The sigmoid gate $\mathbf{g}$ determines which features are active, while the magnitude pathway $\mathbf{m}$ computes the feature strengths. The post-ReLU features $\mathbf{f}$ are truly sparse, while the pre-ReLU features $\mathbf{f}_{\text{pre}}$ retain gradient flow through the sigmoid gate. Reconstruction is performed by $\hat{\mathbf{h}} = W_{\text{dec}}\mathbf{f} + \mathbf{b}_{\text{dec}}$, where $W_{\text{dec}} \in \mathbb{R}^{D \times d}$ has unit-norm rows.

4 Method

SAE-FD operates as a three-stage pipeline within a continual learning loop: (1) training a Gated SAE to decompose model activations into sparse features, (2) training on each task with anchor capture, and (3) applying sparse-aware feature distillation during subsequent tasks. Algorithm 1 provides the complete procedure.

4.1 Stage 1: SAE Training

As shown in Figure 1 (Stage 1), before the continual learning sequence begins, we train a Gated SAE to decompose the base model's internal representations into sparse features. This SAE is trained once per base model and remains frozen throughout the continual learning process.

Activation collection. We collect last-layer MLP activations from the base model on diverse text sources, including classification datasets (AG News, Amazon, Yelp, DBpedia, Yahoo), question answering (BoolQ, ARC), and knowledge benchmarks (MMLU), with up to 2,000 samples per source. For each input, we extract the last non-padding token's MLP output from the final transformer layer, yielding a dataset of d-dimensional activation vectors that captures the model's representation diversity.

Training objective. The SAE is trained with a reconstruction loss and an L1 sparsity penalty:

$$\mathcal{L}_{\text{SAE}} = \|\hat{\mathbf{h}} - \mathbf{h}\|_2^2 + \lambda_{\ell_1} \|\mathbf{f}\|_1 \tag{7}$$

where $\hat{\mathbf{h}} = W_{\text{dec}}\mathbf{f} + \mathbf{b}_{\text{dec}}$ is the reconstruction and $\mathbf{f}$ is the post-ReLU feature vector (Eq. 6). The

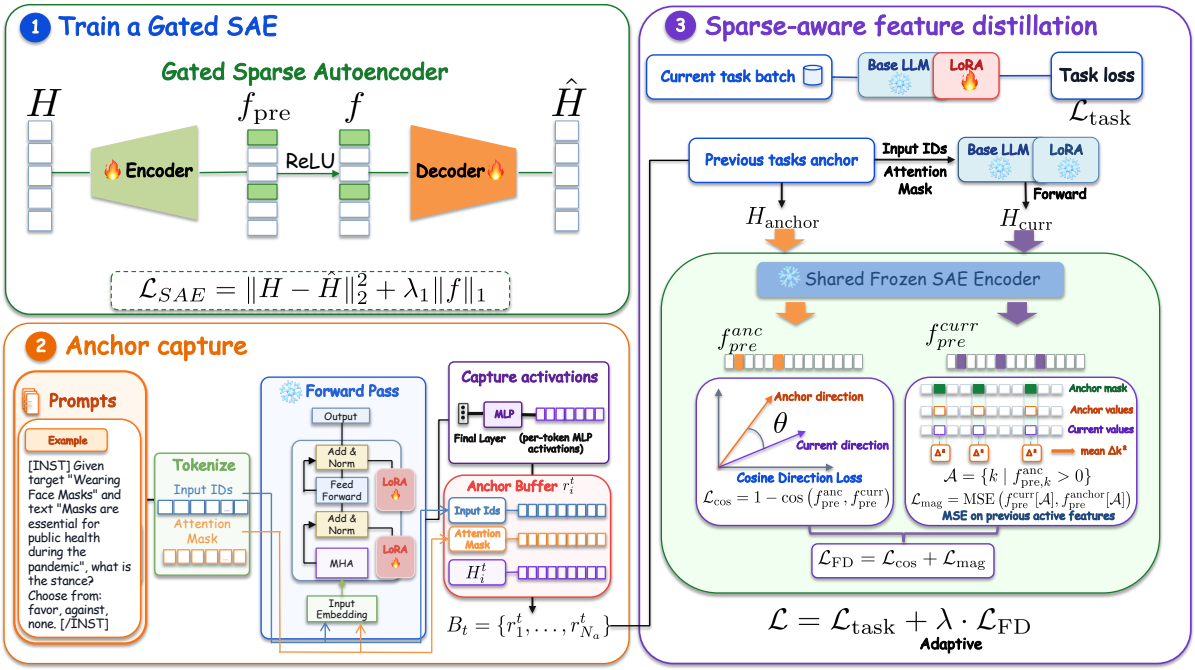

Figure 1: Overview of SAE-FD. **Stage 1**: A Gated SAE is trained to decompose dense activations into sparse features. **Stage 2**: After each task's training, per-token MLP activations on anchor samples are captured and stored. **Stage 3**: Stored anchor activations and recomputed current activations are encoded through the frozen SAE, and the sparse-aware distillation loss combines cosine direction preservation with active-feature magnitude matching, weighted by an adaptive λ.

reconstruction term ensures the SAE faithfully represents the original activations, while the L1 term encourages sparse feature usage, activating only the features relevant to each input. We use d=4096 and D=32768 (8× expansion), with $\lambda_{\ell_1} = 10^{-3}$. Each trained SAE achieves >99% variance explained on held-out activations.

4.2 Stage 2: Task Training and Anchor Capture

As shown in Figure 1 (Stage 2), this stage serves two purposes: fine-tuning $\mathcal{M}$ on each task, and capturing a representation snapshot in SAE feature space that anchors the model's learned knowledge for subsequent distillation. For each task t in the continual learning sequence, we fine-tune $\mathcal{M}$ on $\mathcal{D}_t$ using LoRA (Hu et al., 2022) with the standard language modeling objective:

$$\mathcal{L}_{\text{task}} = -\sum_j \log P_{\mathcal{M}}(y_j \mid x_{<j}) \tag{8}$$

where y_j denotes the target tokens. When $t > 1$, the training loss additionally includes the feature distillation term described in §4.3.

Anchor capture. To provide a reference for subsequent distillation in SAE feature space, we capture a compact snapshot of the model's representations after training on each task. Specifically, we select N_a anchor samples from $\mathcal{D}_t$ and perform a forward pass through $\mathcal{M}$. For each anchor input $x_i = (x_1, \ldots, x_L)$ with attention mask $\mathbf{m}_i$, we extract the per-token MLP output from the last transformer layer, yielding $\mathbf{H}_i \in \mathbb{R}^{L \times d}$. The anchor tuple $(x_i, \mathbf{m}_i, \mathbf{H}_i)$ is stored in float16 for memory efficiency (∼0.8 GB per task). During future training, both $\mathbf{H}_i$ and the current model's activations on x_i are encoded through the frozen SAE $\mathcal{S}$ to compute the distillation loss, enabling the model to be regularized in the decoupled feature space rather than in the raw activation space.

Anchor accumulation. The anchor buffer accumulates across tasks: when training on task t+1, the buffer contains anchors from all tasks $1, \ldots, t$. At each training step, one anchor batch is sampled uniformly at random from the buffer, amortizing the distillation cost across all previous tasks.

4.3 Stage 3: Sparse-Aware Feature Distillation

As shown in Figure 1 (Stage 3), as the model trains on task t+1, its internal representations inevitably drift from those learned for previous tasks. The

goal of this stage is to constrain this drift in SAE feature space, where the decoupled structure allows us to measure and penalize representational changes more precisely than in the raw activation space. Concretely, we encode both the stored anchor activations $\mathbf{H}_a$ and the model's current activations on the same input through the frozen SAE $\mathcal{S}$, and compare the resulting feature vectors. Both encodings use the pre-ReLU features (Eq. 5) to ensure gradient flow through the sigmoid gate; post-ReLU features would zero out gradients for inactive features.

Representation drift in SAE space manifests in two complementary ways: the overall feature direction may rotate, and individual feature magnitudes may shift. We address each with a dedicated loss term.

Cosine direction loss. To prevent the global feature direction from drifting, we penalize the angular deviation between the current feature vector $\mathbf{f}_{\text{pre}}^{\text{curr}}$ produced by encoding the model's current activations and the anchor feature vector $\mathbf{f}_{\text{pre}}^{\text{anc}}$ produced by encoding the stored activations, both through the frozen SAE:

$$\mathcal{L}_{\text{cos}} = 1 - \frac{\mathbf{f}_{\text{pre}}^{\text{curr}} \cdot \mathbf{f}_{\text{pre}}^{\text{anc}}}{\|\mathbf{f}_{\text{pre}}^{\text{curr}}\| \cdot \|\mathbf{f}_{\text{pre}}^{\text{anc}}\|} \tag{9}$$

This loss is computed only on tokens whose anchor feature norm exceeds a threshold (10% of the batch median norm), avoiding numerical instability from near-zero vectors.

Active magnitude loss. While the cosine loss preserves the global direction, it is scale-invariant and cannot prevent individual feature magnitudes from drifting. We therefore add a magnitude loss that applies MSE selectively on features that were active (post-ReLU > 0) in the anchor, directly constraining the features that contributed to the model's previous-task behavior while leaving inactive features free to be recruited for new tasks:

$$\mathcal{L}_{\text{mag}} = \frac{1}{|\mathcal{A}|} \sum_{k \in \mathcal{A}} (f_{\text{pre},k}^{\text{curr}} - f_{\text{pre},k}^{\text{anc}})^2 \tag{10}$$

where $\mathcal{A} = \{k : f_k^{\text{anc}} > 0\}$ is the sparse set of previously active features. By focusing regularization only on these active features, this loss directly leverages the sparse, decoupled structure of SAE representations. Inactive features are left entirely unconstrained, preserving the model's capacity to recruit new features for upcoming tasks.

Combined loss. The feature distillation loss $\mathcal{L}_{\text{FD}} = \mathcal{L}_{\text{cos}} + \mathcal{L}_{\text{mag}}$ is averaged over non-padding tokens in the sampled anchor batch. The cosine term preserves the global direction of the feature representation while the magnitude term ensures individual active features retain their strengths, jointly maintaining both the structure and the scale of previously learned representations.

4.4 Adaptive λ Mechanism

The total training objective for task $t+1$ combines the task loss with the feature distillation loss:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda \cdot \mathcal{L}_{\text{FD}} \tag{11}$$

A key challenge is setting the coefficient λ. A fixed λ is problematic because the magnitudes of $\mathcal{L}_{\text{task}}$ and $\mathcal{L}_{\text{FD}}$ vary substantially across tasks and training steps: $\mathcal{L}_{\text{task}}$ is typically large at the start of a new task and decreases as the model converges, while $\mathcal{L}_{\text{FD}}$ depends on how much the current representations have drifted from the anchors. A fixed λ therefore cannot maintain a consistent balance between learning and regularization throughout training.

We address this with an adaptive mechanism that maintains a target contribution ratio ρ of the distillation loss to the total loss. Given the current losses, the ideal λ satisfying $\rho = \frac{\lambda \mathcal{L}_{\text{FD}}}{\lambda \mathcal{L}_{\text{FD}} + \mathcal{L}_{\text{task}}}$ is:

$$\lambda^* = \frac{\rho}{1-\rho} \cdot \frac{\mathcal{L}_{\text{task}}}{\mathcal{L}_{\text{FD}}} \tag{12}$$

To avoid abrupt changes, λ is updated via exponential moving average with clipping:

$$\lambda_{t+1} = (1-\alpha)\lambda_t + \alpha \cdot \text{clip}(\lambda^*, \lambda_{\min}, \lambda_{\max}) \tag{13}$$

This design produces an intuitive behavior: since $\mathcal{L}_{\text{task}}$ is high when a new task begins, λ automatically starts large, providing strong anti-forgetting protection precisely when the model is most vulnerable to catastrophic forgetting. As the model converges on the new task, $\mathcal{L}_{\text{task}}$ decreases and λ relaxes accordingly, allowing the model to fine-tune without excessive constraint. The mechanism adapts to each task's learning dynamics without manual tuning. Hyperparameter values (ρ, α, $\lambda_{\min}$, $\lambda_{\max}$) are reported in Appendix C.

5 Experiments

We design experiments to answer the following research questions: **RQ1**: Does SAE-FD effectively reduce catastrophic forgetting compared to existing

Algorithm 1 SAE-FD Continual Learning Pipeline

Require: Task sequence $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$, model $\mathcal{M}$, trained SAE $\mathcal{S}$, anchor count N_a

$\mathcal{A} \leftarrow \emptyset$ ▷ Anchor buffer
for $t = 1, \ldots, T$ **do**
  for each training step on $\mathcal{D}_t$ **do**
    $\mathcal{L}_{\text{task}} \leftarrow \text{CrossEntropy}(\mathcal{M}(x), y)$
    if $\mathcal{A} \neq \emptyset$ **then**
      Sample $(x_a, \mathbf{m}_a, \mathbf{H}_a)$ from $\mathcal{A}$
      $\mathbf{H}_{\text{curr}} \leftarrow \text{MLP}_{\text{last}}(\mathcal{M}, x_a)$
      $\mathbf{f}_{\text{pre}}^{\text{curr}} \leftarrow \mathcal{S}.\text{encode_pre_relu}(\mathbf{H}_{\text{curr}})$
      $\mathbf{f}_{\text{pre}}^{\text{anc}} \leftarrow \mathcal{S}.\text{encode_pre_relu}(\mathbf{H}_a)$
      $\mathcal{L}_{\text{FD}} \leftarrow \mathcal{L}_{\text{cos}} + \mathcal{L}_{\text{mag}}$ ▷ Eq. 7–8
      Update λ via EMA ▷ Eq. 10
      $\mathcal{L} \leftarrow \mathcal{L}_{\text{task}} + \lambda \cdot \mathcal{L}_{\text{FD}}$
    else
      $\mathcal{L} \leftarrow \mathcal{L}_{\text{task}}$
    end if
    Update LoRA parameters via $\nabla \mathcal{L}$
  end for
  Anchor capture:
  Select N_a samples from $\mathcal{D}_t$
  for each anchor sample x_i **do**
    $\mathbf{H}_i \leftarrow \text{MLP}_{\text{last}}(\mathcal{M}, x_i)$ ▷ Per-token
    Store $(x_i, \mathbf{m}_i, \mathbf{H}_i^{\text{fp16}})$ in $\mathcal{A}$
  end for
end for

CL methods across different benchmarks? **RQ2**: Does the approach generalize across model architectures? **RQ3**: How do the individual components (loss design, adaptive λ, SAE decomposition) contribute to performance? **RQ4**: How sensitive is SAE-FD to its key hyperparameters?

5.1 Setup

Benchmarks. We primarily evaluate on **TRACE** (Wang et al., 2023b), a challenging CL benchmark that requires sequential adaptation across 8 heterogeneous tasks—from stance detection and financial sentiment to code completion and German text simplification—with 5,000 training and 500 test samples per task. The diversity of task types (classification, generation, reasoning, multilingual) makes TRACE particularly suited for evaluating whether a CL method can preserve knowledge across dissimilar domains. We additionally report results on a **Standard CL** 4-task text classification benchmark (Wang et al., 2025, 2023a) in Appendix I. Detailed task descriptions, example prompts, and per-task evaluation metrics are provided in Appendix A and B.

Models and baselines. We evaluate across LLaMA-2-7B-Chat (Touvron et al., 2023), Vicuna-7B-v1.5 (Chiang et al., 2023), and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), each with a separately trained Gated SAE. Baselines include SeqLoRA, EWC (Kirkpatrick et al., 2017), O-LoRA (Wang et al., 2023a), GORP (Wang et al., 2025), TreeLoRA (Qian et al., 2025), L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), GEM (Lopez-Paz and Ranzato, 2017), OGD (Farajtabar et al., 2020), HiDe-PET (Wang et al., 2024), LwF (Li and Hoiem, 2017), SEEKR (He et al., 2024), PP (Razdaibiedina et al., 2023), LFPT5 (Qin and Joty, 2022), and DER++ (Buzzega et al., 2020). We use published results where available; our reproduced baselines follow the official TRACE configuration. Full training details and hyperparameters are in Appendix C.

5.2 Main Results

SAE-FD achieves the best plasticity-stability tradeoff among all compared methods. Table 1 shows that SAE-FD reaches 52.70% AA with only -0.46% BWT on LLaMA-2-7B-Chat. Compared to GORP, the strongest published regularization-based method, SAE-FD improves accuracy by +2.3% and reduces forgetting by 34% (-0.46 vs -0.70 BWT). The gap is more striking against other LoRA-based methods: EWC, O-LoRA, and TreeLoRA all show BWT worse than -3%, indicating that weight-space and gradient-space regularization struggle to prevent forgetting on this diverse benchmark. This validates our core hypothesis that sparse feature space provides a more effective regularization target than dense spaces where task features are superimposed. A per-task BWT breakdown (Table 10) shows that five of seven tasks exhibit $|\text{BWT}| < 2\%$, and the full accuracy matrix (Table 11) confirms stable performance across the entire 8-task sequence.

SAE-FD generalizes to standard classification CL. To evaluate beyond the heterogeneous TRACE benchmark, we test on a 4-task text classification benchmark across 3 random task orders (Table 14). SAE-FD achieves 88.25% AA with -0.89% BWT, reducing forgetting by $5.3\times$ compared to SeqLoRA (-4.71%), and shows low variance across orderings (AA ranges from 87.90% to 88.55%), suggesting robustness to task ordering.

Method	AA↑	BWT↑
Prompt-based		
PP (Razdaibiedina et al., 2023)	29.41	−5.79
L2P (Wang et al., 2022b)	36.23	−8.25
DualPrompt (Wang et al., 2022a)	37.69	−8.03
LFPT5 (Qin and Joty, 2022)	38.67	−11.43
LoRA / Regularization / Gradient		
SeqLoRA	34.30	−18.50
GEM (Lopez-Paz and Ranzato, 2017)	40.08	−6.77
HiDe-PET (Wang et al., 2024)	41.60	−7.12
LwF (Li and Hoiem, 2017)	41.86	−6.50
OGD (Farajtabar et al., 2020)	42.09	−8.06
EWC (Kirkpatrick et al., 2017)	42.36	−5.97
O-LoRA (Wang et al., 2023a)	42.78	−7.16
TreeLoRA (Qian et al., 2025)	43.52	−3.46
GORP (Wang et al., 2025)	50.40	−0.70
SAE-FD (ours)	**52.70**	**−0.46**

Table 1: TRACE benchmark results on LLaMA-2-7B-Chat. AA and BWT are reported in %. SAE-FD achieves the best AA and BWT among all compared methods.

Model	Method	AA↑	BWT↑
LLaMA-2 -7B	EWC (Kirkpatrick et al., 2017)	42.36	−5.97
	O-LoRA (Wang et al., 2023a)	42.78	−7.16
	TreeLoRA (Qian et al., 2025)	43.52	−3.46
	GORP (Wang et al., 2025)	50.40	−0.70
	SAE-FD (ours)	**52.70**	**−0.46**
Vicuna -7B	LwF (Li and Hoiem, 2017)	41.19	−5.54
	EWC (Kirkpatrick et al., 2017)	41.88	−15.57
	O-LoRA (Wang et al., 2023a)	43.42	−6.27
	SAE-FD (ours)	**53.50**	**−0.69**
Mistral -7B	O-LoRA (Wang et al., 2023a)	52.02	−8.13
	EWC (Kirkpatrick et al., 2017)	52.45	−5.98
	TreeLoRA (Qian et al., 2025)	54.77	−3.77
	SAE-FD (ours)	**57.80**	**−1.63**

Table 2: Multi-model TRACE results compared with published baselines. Each model uses a separately trained SAE. AA and BWT are reported in %. SAE-FD consistently achieves the best performance across all three architectures. Full baseline tables are provided in Appendix G.

5.3 Multi-Model Generalization

SAE-FD consistently outperforms published baselines across architectures. Table 2 compares SAE-FD against the strongest published baselines for each model. On LLaMA-2, SAE-FD (52.70%) outperforms GORP (50.40%), the strongest published regularization-based method, by +2.3% AA with 34% less forgetting. On Vicuna, SAE-FD (53.50%) surpasses O-LoRA (43.42%) by +10.1% AA and achieves the best BWT among all methods (−0.69%), including full fine-tuning baselines. On Mistral, SAE-FD (57.80%) surpasses TreeLoRA (54.77%), the strongest published result, by +3.0% AA with 2.3× less forgetting. These consistent improvements across three architectures, each using a separately trained SAE with no method-level changes, demonstrate that the sparse feature decomposition principle generalizes beyond a single model family. Notably, the gains are largest on Vicuna (+10.1% AA over O-LoRA), where dense-space baselines degrade severely, suggesting that SAE-based regularization is particularly effective when the base model's representations are more susceptible to forgetting.

5.4 Ablation Studies

To understand the contribution of each design choice, we conduct ablation studies on TRACE (Table 3).

Loss components reveal complementary roles. To examine whether both loss terms are necessary, we train SAE-FD with each component alone. As shown in Table 3, cosine-only loss achieves 51.29% AA with −0.51% BWT—preserving feature directions but lacking magnitude control leads to lower accuracy. Magnitude-only loss improves AA to 52.57% but BWT degrades to −0.84%, as matching magnitudes without direction constraints allows the overall representation to rotate. The combined loss achieves the best on both metrics (52.70% AA, −0.46% BWT), confirming that direction and magnitude preservation address complementary aspects of representation drift.

Adaptive λ is critical for strong anti-forgetting. To evaluate the adaptive mechanism, we compare it against a fixed λ=1.0. As shown in Table 3, the fixed setting yields −2.38% BWT, comparable to other regularization methods, while the adaptive mechanism achieves −0.46%, a 5.2× improvement. The dynamics (Appendix D) reveal intuitive behavior: λ starts at the maximum when beginning a new task and decays at task-dependent rates, e.g., short tasks like FOMC (3 epochs) decay rapidly (10.0 → 0.20) while longer tasks like MeetingBank (7 epochs) maintain elevated λ (10.0 → 4.80).

SAE decomposition outperforms raw activation matching. To isolate the benefit of operating in SAE feature space, we compare SAE-FD against raw MSE distillation on the original 4096-dimensional MLP activations. As shown in Table 3, SAE-FD consistently outperforms raw MSE: on Vi-

Ablation	Setting	AA↑	BWT↑
Loss (LLaMA-2)	Cosine only	51.29	-0.51
	Magnitude only	52.57	−0.84
	Both	**52.70**	**−0.46**
λ (LLaMA-2)	Fixed λ=1.0	51.41	−2.38
	Adaptive	**52.70**	**−0.46**
SAE vs. MSE (Vicuna)	Raw MSE	51.31	−0.96
	SAE-FD	**53.50**	**−0.69**
SAE vs. MSE (Mistral)	Raw MSE	56.80	−3.33
	SAE-FD	**57.13**	**−1.47**

Table 3: Ablation studies on TRACE. AA and BWT are reported in %. Each component contributes to the overall performance, and SAE-based distillation consistently outperforms raw MSE matching.

cuna by +2.19% AA (−0.69 vs −0.96 BWT), and on Mistral by +1.86% BWT (−1.47 vs −3.33), confirming that the decoupled SAE space provides a more effective regularization target than the entangled dense space. We additionally verify that the frozen SAE maintains faithful reconstruction quality (>90% variance explained) throughout the 8-task sequence (Figure 2), and that the computational overhead is approximately 1.85× SeqLoRA with constant cost per step regardless of buffer size (Table 13).

5.5 Hyperparameter Sensitivity

We evaluate the sensitivity of SAE-FD to its two key hyperparameters: the number of anchor samples per task (N_a) and the adaptive λ target ratio (ρ). All experiments are conducted on TRACE with LLaMA-2-7B-Chat.

Anchor count N_a. Table 4 shows results for $N_a \in \{50, 100, 200, 400\}$. Increasing the anchor count consistently improves BWT (from −1.45% at N_a=50 to −0.22% at N_a=400), as more anchor samples provide a more representative snapshot of previous-task representations. AA remains stable across all settings (51.65–53.45%), and all configurations substantially outperform SeqLoRA (AA=38.21%). We use N_a=200 as the default, balancing storage cost (∼0.8 GB/task) and anti-forgetting effectiveness.

Target ratio ρ. Table 5 shows results for $\rho \in \{0.05, 0.10, 0.15, 0.25, 0.35\}$. The method is robust across a wide range of ρ values: AA ranges from 51.50% to 54.35% and BWT from −0.94% to +0.63%, all substantially better than SeqLoRA. We use ρ=0.15 as the default, which provides a good tradeoff without overly constraining plasticity.

Anchor/task	AA (%)↑	BWT (%)↑
50	52.43	−1.45
100	51.65	−0.66
200 (default)	**52.70**	**−0.46**
400	53.45	−0.22

Table 4: Sensitivity to anchor count N_a on TRACE (LLaMA-2-7B-Chat). Increasing N_a consistently improves BWT while AA remains stable, indicating that a more representative anchor snapshot strengthens anti-forgetting without hurting new-task learning.

Target ratio ρ	AA (%)↑	BWT (%)↑
0.05	51.50	−0.94
0.10	52.09	+0.06
0.15 (default)	**52.70**	**−0.46**
0.25	53.07	−0.51
0.35	54.35	+0.63

Table 5: Sensitivity to adaptive λ target ratio ρ on TRACE (LLaMA-2-7B-Chat). The method is robust across a wide range; all settings outperform SeqLoRA (AA=38.21%).

Standard CL benchmark results and the full table with published baselines are provided in Appendix I.

6 Conclusion

We introduced SAE-FD, a continual learning method that anchors representations in the sparse feature space of a pre-trained Sparse Autoencoder. By decomposing dense activations into a sparse, decoupled feature space, SAE-FD enables more targeted preservation of previously learned knowledge while maintaining plasticity for new tasks. Our approach addresses the key technical challenges of operating in SAE feature space through a sparse-aware distillation loss on pre-ReLU features with active-feature masking and an adaptive λ mechanism that automatically balances plasticity and stability. SAE-FD achieves strong results on TRACE and a 4-task classification benchmark, outperforming compared methods across three model families. Our work connects mechanistic interpretability with continual learning: the sparse, disentangled representations developed for understanding neural networks also serve as an effective inductive bias for controlling them. Future work could explore multi-layer SAE features, scaling to longer task sequences, and leveraging SAE feature interpretability for task-aware regularization.

Limitations

SAE-FD requires a pre-trained SAE per base model, adding a one-time training cost that limits its applicability to architectures where SAEs are unavailable or perform poorly. The method stores float16 MLP activations for anchor samples ($\sim$0.8 GB per task), growing linearly with task count; for very long task sequences, compression strategies may be needed. TRACE evaluates on a single fixed task order, and while multi-model consistency suggests robustness, the Standard CL benchmark shows moderate variance across orderings (SAE-FD AA ranges from 87.90% to 88.55%). Finally, published baselines use different experimental settings (e.g., GORP uses 1 epoch/task vs. the official multi-epoch schedule, and SEEKR uses full fine-tuning vs. our LoRA), so cross-method comparisons should be interpreted with these caveats in mind.

References

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Alex Tamkin, Karina Nguyen, Brayden McLean, and 5 others. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. 2020. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems*, volume 33.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. Blog post.

Shuting Cui, Ying Sun, Yuting Zhang, Qingxin Meng, and Hengshu Zhu. 2026. LLM-enhanced career knowledge graph understanding for job mobility prediction. *ACM Transactions on Management Information Systems*.

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

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. 2020. Orthogonal gradient descent for continual learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3762–3773. PMLR.

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

Jinghan He, Haiyun Guo, Kuan Zhu, Zihan Zhao, Ming Tang, and Jinqiao Wang. 2024. SEEKR: Selective attention-guided knowledge retention for continual learning of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3254–3266, Miami, Florida, USA. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947.

Yan-Shuo Liang and Wu-Jun Li. 2024. InfLoRA: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23638–23647.

Huanxuan Liao, Shizhu He, Yupu Hao, Jun Zhao, and Kang Liu. 2025. DATA: Decomposed attention-based task adaptation for rehearsal-free continual learning. *arXiv preprint arXiv:2502.11482*.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30.

Yu-Yang Qian, Yuan-Ze Xu, Zhen-Yu Zhang, Peng Zhao, and Zhi-Hua Zhou. 2025. TreeLoRA: Efficient continual learning via layer-wise LoRAs guided by a hierarchical gradient-similarity tree. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, pages 50066–50085.

Fuli Qiao and Mehrdad Mahdavi. 2025. Merge before forget: A single LoRA continual learning via continual merging. *arXiv preprint arXiv:2512.23017*.

Chengwei Qin and Shafiq Joty. 2022. LFPT5: A unified framework for lifelong few-shot language learning based on prompt tuning of T5. In *International Conference on Learning Representations*.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. 2024. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. 2023. Progressive prompts: Continual learning for language models. In *International Conference on Learning Representations*.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameber, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, and 3 others. 2024. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Chenxu Wang, Yilin Lyu, Zicheng Sun, and Liping Jing. 2025. Continual gradient low-rank projection fine-tuning for LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14815–14829, Vienna, Austria. Association for Computational Linguistics.

Liyuan Wang, Jingyi Xie, Xingxing Zhang, Hang Su, and Jun Zhu. 2024. HiDe-PET: Continual learning via hierarchical decomposition of parameter-efficient tuning. *arXiv preprint arXiv:2407.05229*.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2023a. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, Singapore. Association for Computational Linguistics.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023b. TRACE: A comprehensive benchmark for continual learning in large language models. *arXiv preprint arXiv:2310.06762*.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022a. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022b. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 139–149.

Shuo Yang, Kun-Peng Ning, Yu-Yang Liu, Jia-Yu Yao, Yong-Hong Tian, Yi-Bing Song, and Li Yuan. 2025. Is parameter collision hindering continual learning in LLMs? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4243–4259, Abu Dhabi, UAE. Association for Computational Linguistics.

Mingxu Zhang, Yuhan Li, Lujundong Li, Dazhong Shen, Hui Xiong, and Ying Sun. 2026. Slim: Sparse latent steering for interpretable and property-directed llm-based molecular editing. *arXiv preprint arXiv:2605.10831*.

Mingxu Zhang, Dazhong Shen, and Ying Sun. 2025a. AtomDisc: An atom-level tokenizer that boosts molecular LLMs and reveals structure–property associations. *arXiv preprint arXiv:2512.03080*.

Mingxu Zhang, Dazhong Shen, Qi Zhang, and Ying Sun. 2025b. ChemATP: A training-free chemical reasoning framework for large language models. *arXiv preprint arXiv:2512.19240*.

Yunan Zhang, Shuoran Jiang, Mengchen Zhao, Yuefeng Li, Yang Fan, Xiangping Wu, and Qingcai Chen. 2025c. GeRe: Towards efficient anti-forgetting in continual learning of LLM via general samples replay. *arXiv preprint arXiv:2508.04676*.

Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. 2024. SAPT: A shared attention framework for parameter-efficient continual learning of large language models. In *Proceedings of the*

62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 11641–11661, Bangkok, Thailand. Association for Computational Linguistics.

A TRACE Benchmark Details

TRACE (Wang et al., 2023b) is a comprehensive continual learning benchmark for LLMs consisting of 8 heterogeneous tasks trained in a fixed order. The benchmark is designed to test continual learning across diverse task types, including classification, generation, code completion, reasoning, and multilingual tasks. Each task provides 5,000 training samples and 500 test samples. Table 6 summarizes the tasks, and Table 7 provides representative prompt-output examples.

All tasks use the LLaMA-2 instruction format: `[INST] {prompt} [/INST]`, where the task-specific prompt is wrapped in instruction tags. During evaluation, the model generates up to 256 tokens with temperature 0.1.

B Evaluation Metrics

TRACE uses task-specific evaluation metrics reflecting the diversity of its 8 tasks, following the official evaluation protocol from Wang et al. (2023b). Classification tasks (C-STANCE, FOMC, NumGLUE-cm, NumGLUE-ds) use strict string match accuracy: the model's generated output must exactly match the reference answer. ScienceQA uses first-character match accuracy, extracting the answer letter from the beginning of the generated text. Code completion (Py150) uses the fuzzywuzzy string similarity ratio (0–100, normalized to 0–1), which measures the edit distance between the generated and reference code. Summarization tasks (MeetingBank, 20Minuten) use ROUGE-L F1 score computed via py-rouge, measuring the longest common subsequence between the generated and reference summaries. All evaluations use temperature 0.1 sampling with a maximum of 256 new tokens and a maximum prompt length of 512 tokens.

C Hyperparameters and Training Details

Table 8 lists the complete hyperparameter settings for all experiments. LoRA is applied to the query, key, value, and output projection matrices of all attention layers. The per-task epoch schedule follows the official TRACE configuration, with longer schedules assigned to more complex tasks. For SAE-FD, 200 anchor samples are captured per task and stored in float16, totaling approximately 5.6 GB across all 7 anchor sets (tasks 1–7). The adaptive λ mechanism uses a target contribution ratio of 15%, meaning the distillation loss is maintained at approximately 15% of the total training loss.

D Adaptive λ Dynamics

Table 9 shows the adaptive λ values at the start and end of each task's training for LLaMA-2-7B-Chat. The mechanism exhibits intuitive behavior aligned with forgetting risk: λ is initialized near its maximum when a new task begins (high task loss drives up the ideal λ) and decays as the model converges. Notably, task-dependent decay rates emerge automatically: short tasks like FOMC (3 epochs) see rapid decay to the minimum ($10.0 \rightarrow 0.20$), while longer and more complex tasks like MeetingBank (7 epochs) maintain elevated λ throughout ($10.0 \rightarrow 4.80$), reflecting sustained forgetting risk from extended training.

E Per-Task BWT Breakdown

Table 10 shows the per-task BWT for SAE-FD on TRACE (LLaMA-2-7B-Chat). Five of seven previously learned tasks exhibit $|\text{BWT}| < 2\%$, with positive transfer observed on FOMC (+0.8%) and NumGLUE-cm (+2.4%). The largest degradation occurs on ScienceQA (-3.6%), a multi-step reasoning task requiring chain-of-thought generation, suggesting that compositional reasoning features may be harder to preserve than simpler classification or generation features.

F Full Accuracy Matrix

Table 11 presents the full accuracy matrix for SAE-FD on TRACE (LLaMA-2-7B-Chat). Each row shows performance on all tasks seen so far after training on the corresponding task. The matrix demonstrates remarkable stability: most previously learned tasks maintain their performance throughout the entire 8-task sequence, with only minor fluctuations. For example, FOMC accuracy remains in the 69.8–72.4% range from task 2 through task 8, and MeetingBank stays within 60.5–61.2% from task 3 onward.

G Published Baselines by Model

Table 12 provides comprehensive comparisons with all published baselines for Mistral-7B and Vicuna-7B. These baselines are reported from the original papers using their respective experimental configurations. SAE-FD outperforms all published LoRA-based baselines on both models, and on Vicuna-7B

Order	Task	Domain	Metric
T1	C-STANCE	Stance detection on COVID-related claims	Accuracy
T2	FOMC	Financial policy sentiment from Fed minutes	Accuracy
T3	MeetingBank	Summarization of meeting transcripts	ROUGE-L
T4	Py150	Python code completion	Similarity
T5	ScienceQA	Science multiple-choice QA with reasoning	Accuracy
T6	NumGLUE-cm	Commonsense arithmetic word problems	Accuracy
T7	NumGLUE-ds	Numerical data science reasoning	Accuracy
T8	20Minuten	German news text simplification	ROUGE-L

Table 6: TRACE benchmark task overview. Tasks are trained in fixed order T1–T8.

Task	Prompt (abbreviated)	Output
C-STANCE	Given the target "Wearing Face Masks" and the text "Masks are essential for public health during the pandemic", what is the stance? Choose from: favor, against, none.	favor
FOMC	Classify the following sentence from FOMC minutes as hawkish, dovish, or neutral: "The Committee decided to raise the target range for the federal funds rate by 25 basis points."	hawkish
Meeting-Bank	Summarize the key decisions from the following meeting transcript: "The board reviewed the quarterly budget report ..."	The board approved the revised budget ...
Py150	Complete the following Python code: `def fibonacci(n): if n <= 1:`	`return n return fibonacci(n-1) + fibonacci(n-2)`
ScienceQA	Question: Which property do these two objects have in common? Select the best answer. (A) smooth (B) scratchy (C) blue	A
NumGLUE-cm	Sam had 9 dimes in his bank. His dad gave him 7 more dimes. How many dimes does Sam have now?	16
NumGLUE-ds	There are 7 dogwood trees currently in the park. Park workers will plant 3 more dogwood trees today. How many dogwood trees will the park have?	10
20Minuten	Vereinfache den folgenden Text: "Die Regierung hat beschlossen, die Massnahmen zur Bekämpfung der Inflation zu verschärfen ..."	Die Regierung verschärft die Massnahmen gegen die Inflation ...

Table 7: Representative prompt-output examples for each TRACE task.

achieves competitive accuracy with SEEKR-1% while using parameter-efficient LoRA instead of full fine-tuning.

Parameter	Value
LoRA	
Rank / Alpha / Dropout	8 / 32 / 0.1
Target modules	q, k, v, o_proj
Training	
Learning rate	1e-4 (const. + 10% warmup)
Optimizer	AdamW (wd 0.0)
Batch (effective)	4×8 accum = 32
Per-task epochs	[5,3,7,5,3,5,5,7]
Max seq. length	1024
Train / eval samples	5000 / 500 per task
SAE-FD	
Anchors per task	200
Target ρ / EMA α	0.15 / 0.05
$\lambda_{\min}$ / $\lambda_{\max}$	0.2 / 10.0
SAE Training	
Dimensions ($d \to D$)	$4096 \to 32768$
Type	Gated SAE
Training loss	MSE + L1 ($\lambda_{\ell_1} = 10^{-3}$)
Optimizer	AdamW (lr 3e-4, cosine decay)
Epochs / Batch size	30 / 128
Variance explained	$>$99%
Evaluation	
Max prompt / new tokens	512 / 256
Temperature / do_sample	0.1 / True

Table 8: Complete hyperparameter settings.

Task	λ_{start}	λ_{end}
T2 (FOMC, 3 epochs)	10.0	0.20
T3 (MeetingBank, 7 epochs)	10.0	4.80
T4 (Py150, 5 epochs)	10.0	0.97
T5 (ScienceQA, 3 epochs)	7.5	3.3
T6 (NumGLUE-cm, 5 epochs)	1.6	0.20
T7 (NumGLUE-ds, 5 epochs)	4.9	0.44

Table 9: Adaptive λ dynamics for LLaMA-2-7B-Chat on TRACE.

Task	Peak	Final	BWT
C-STANCE	48.8	48.0	-0.8
FOMC	69.8	70.6	+0.8
MeetingBank	61.2	60.9	-0.3
Py150	48.4	48.3	-0.1
ScienceQA	82.4	78.8	-3.6
NumGLUE-cm	38.3	40.7	+2.4
NumGLUE-ds	53.5	51.7	-1.8
Average			$\mathbf{-0.46}$

Table 10: Per-task BWT for SAE-FD on TRACE (LLaMA-2-7B-Chat). Values in %.

	C-ST	FOMC	MB	Py150	SciQA	N-cm	N-ds	20Min
After T1	48.8	—	—	—	—	—	—	—
After T2	45.8	69.8	—	—	—	—	—	—
After T3	48.2	71.0	61.2	—	—	—	—	—
After T4	48.0	71.8	60.6	48.4	—	—	—	—
After T5	46.4	71.4	60.5	48.5	82.4	—	—	—
After T6	47.0	72.4	60.5	47.7	78.8	38.3	—	—
After T7	46.4	70.6	60.8	48.4	77.2	34.6	53.5	—
After T8	48.0	70.6	60.9	48.3	78.8	40.7	51.7	22.5

Table 11: Full accuracy matrix (%) for SAE-FD on TRACE (LLaMA-2-7B-Chat). Each row shows performance on all tasks seen so far after training on that row's task.

Model	**Method**	**AA↑**	**BWT↑**
Mistral -7B	SeqLoRA	46.94	−11.41
	L2P (Wang et al., 2022b)	49.32	−5.34
	DualPrompt (Wang et al., 2022a)	51.14	−6.13
	HiDe-PET (Wang et al., 2024)	51.81	−6.25
	O-LoRA (Wang et al., 2023a)	52.02	−8.13
	GEM (Lopez-Paz and Ranzato, 2017)	52.32	−6.01
	EWC (Kirkpatrick et al., 2017)	52.45	−5.98
	TreeLoRA (Qian et al., 2025)	54.77	−3.77
	SAE-FD (ours)	**57.80**	**−1.63**
Vicuna -7B	LwF (Li and Hoiem, 2017)	41.19	−5.54
	EWC (Kirkpatrick et al., 2017)	41.88	−15.57
	O-LoRA (Wang et al., 2023a)	43.42	−6.27
	DER++ (1%) (Buzzega et al., 2020)	49.01	−9.04
	SEEKR (1%) (He et al., 2024)	55.78	−2.64
	SAE-FD (ours)	**53.50**	**−0.69**

Table 12: Full TRACE baseline comparisons for Mistral-7B and Vicuna-7B. AA and BWT are reported in %. SAE-FD achieves the best BWT on both models.

H Computational Overhead

To quantify the computational overhead of SAE-FD, we compare per-task training time and performance against SeqLoRA and SeqLoRA+ER on Vicuna-7B (5,000 samples/task, identical epoch schedule, single NVIDIA H100 GPU). Table 13 reports training time (relative to SeqLoRA), peak scores (immediately after training on each task), final retained scores (after all 8 tasks), and forgetting (Drop = Final − Peak).

For T1 (C-STANCE), SAE-FD has zero overhead (1.00×) because no anchor buffer exists yet. From T2 onward, SAE-FD adds one forward pass on a sampled anchor batch per training step, resulting in consistent ∼2× overhead. This overhead does not grow with the number of previous tasks, as SAE-FD samples a single fixed-size anchor batch regardless of buffer size. The total overhead is 1.85× relative to SeqLoRA. SeqLoRA+ER incurs additional overhead from replaying previous-task data, which grows as the replay buffer expands with each task.

All three methods achieve comparable peak performance on each task, confirming that neither feature distillation nor replay impedes new-task learning. The critical difference is retention: SeqLoRA catastrophically forgets early tasks (MeetingBank: 62.6% → 16.0%), SeqLoRA+ER partially mitigates this (60.3% → 56.6%), while SAE-FD retains nearly all peak performance (64.2% → 62.2%). SAE-FD achieves the best overall AA (53.4%) and BWT (−0.7%), outperforming SeqLoRA+ER (49.7% AA, −3.8% BWT) at 1.85× the cost of SeqLoRA. Additionally, the frozen SAE adds a fixed 256 MB storage cost, and per-task anchor activations add ∼0.8 GB per task in float16.

I Standard CL Benchmark

Table 14 presents results on a 4-task text classification benchmark, averaged over 3 random task orders. SAE-FD achieves 88.25% AA with −0.89% BWT. Our results exceed published baselines (SAPT-LoRA 81.1%, GORP 78.6%), though the gap between our SeqLoRA (85.6%) and published baselines suggests differences in evaluation protocol, so cross-paper comparisons should be interpreted with caution.

J SAE Reconstruction Quality Under Continual Fine-Tuning

A key concern with SAE-FD is whether the frozen SAE remains a faithful decomposition of model activations as LoRA fine-tuning progressively shifts the representations across tasks. If the SAE's reconstruction quality degrades substantially, the feature distillation signal may become unreliable. To investigate this, we evaluate the frozen Vicuna-7B SAE's reconstruction quality on model activations collected after each of the 8 TRACE tasks, using 500 random samples per evaluation.

Figure 2 tracks two key metrics across the 8-task sequence: variance explained (proportion of activation variance captured by the SAE reconstruction) and cosine similarity between original and reconstructed activations. Note that the SAE achieves >99% variance explained on its own held-out training data (diverse general text); the lower starting point of 94.65% on the base model reflects a domain gap between the SAE's training distribution and the TRACE benchmark data, which includes specialized domains such as Python code, financial text, and German text that are underrepresented in the SAE training corpus. From this baseline, variance explained decreases gradually to 90.29% after task 8, with only 4.36 percentage points of additional degradation attributable to LoRA fine-tuning across the entire 8-task sequence. Cosine similarity remains above 0.96 throughout, indicating that the SAE feature space remains structurally intact.

These results justify the use of a frozen SAE for feature distillation: LoRA fine-tuning introduces modest representational shifts that reduce reconstruction fidelity only marginally. The SAE continues to provide a meaningful sparse decomposition of model activations even after 8 sequential tasks, supporting the reliability of the distillation signal throughout the continual learning process.

	Time (×Seq)			Peak (%)			Final (%)			Drop		
Task	**Seq**	**+ER**	**FD**	**Seq**	**+ER**	**FD**	**Seq**	**+ER**	**FD**	**Seq**	**+ER**	**FD**
T1 C-ST	1.00	1.00	1.00	51.0	51.6	47.4	43.4	48.0	48.2	−7.6	−3.6	+0.8
T2 FOMC	1.00	1.03	2.07	70.8	70.0	71.4	49.6	67.3	71.6	−21.2	−2.6	+0.2
T3 MB	1.00	1.06	1.97	62.6	60.3	64.2	16.0	56.6	62.2	−46.6	−3.7	−2.0
T4 Py150	1.00	1.10	2.04	46.6	45.1	45.3	36.5	42.7	45.4	−10.1	−2.4	+0.1
T5 SciQA	1.00	0.87	1.46	80.0	88.4	78.4	71.4	79.8	75.4	−8.6	−8.6	−3.0
T6 N-cm	1.00	1.23	2.08	44.4	34.6	46.9	33.3	32.1	46.9	−11.1	−2.5	+0.0
T7 N-ds	1.00	1.28	2.08	54.2	55.1	58.2	51.4	51.7	57.2	−2.8	−3.4	−1.0
T8 20Min	1.00	1.24	1.96	19.9	19.5	21.1	19.9	19.5	21.1	+0.0	+0.0	+0.0
Overall	**1.00**	**1.12**	**1.85**	**53.7**	**53.1**	**54.1**	**40.2**	**49.7**	**53.4**	**−15.4**	**−3.8**	**−0.7**

Table 13: Per-task training time, peak scores, final retained scores, and forgetting on TRACE (Vicuna-7B). Seq = SeqLoRA, +ER = SeqLoRA+ER, FD = SAE-FD.

Method	**AA↑**	**BWT↑**
Published baselines (LLaMA-2-7B)		
O-LoRA (Wang et al., 2023a)	76.1	—
N-LoRA (Yang et al., 2025)	77.6	—
GORP (Wang et al., 2025)	78.6	—
InfLoRA (Liang and Li, 2024)	79.6	—
DATA (Liao et al., 2025)	79.8	—
SLAO (Qiao and Mahdavi, 2025)	80.4	—
SAPT-LoRA (Zhao et al., 2024)	81.1	—
Our results (LLaMA-2-7B, 3 orders)		
SeqLoRA	85.62	−4.71
SAE-FD	**88.25**	**−0.89**

Table 14: Standard CL benchmark results (4 tasks, LLaMA-2-7B-Chat). AA and BWT are reported in %.

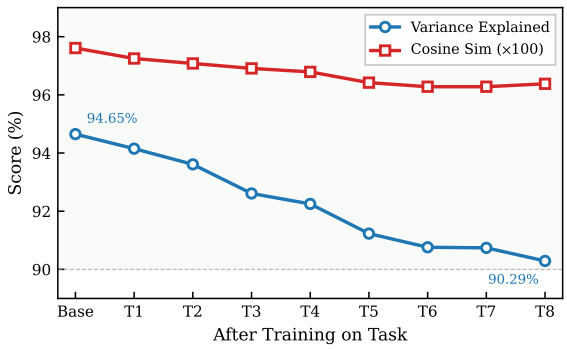

Figure 2: Frozen SAE reconstruction quality on Vicuna-7B activations after each TRACE task. Variance explained decreases gradually from 94.65% to 90.29% (−4.36%) over 8 tasks. Cosine similarity (scaled ×100) remains above 96% throughout, confirming that the frozen SAE provides a reliable feature decomposition under continual LoRA fine-tuning.