

SAE-LOCKED LORA: EDIT-SPECIFIC SPARSE-FEATURE GRADIENT ROUTING FOR LOW-INTERFERENCE KNOWLEDGE EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge edits should change a requested behavior without perturbing unrelated behavior, yet a low-rank adapter can alter broad internal directions despite updating few parameters. We introduce **SAE-LOCKED LORA**, which uses a frozen sparse autoencoder as an edit-specific coordinate system for LoRA optimization. Normalized edit-loss gradients select the smallest bounded latent set covering a target attribution mass; only the edit gradient is routed through this fixed set, while preservation gradients use a separate unmasked pass. On CounterFact, the method retains comparable success to vanilla LoRA (99.6 versus 99.8), improves locality (89.5 versus 63.7), and reduces KL drift (0.16 versus 0.65). Under an objective-matched comparison it gains 7.5 locality points over PCA (95% CI $[6.7, 8.3]$). It reaches 49.8 exact match and 28.5 micro `correct_flip` on matched ReCoE and retains 91.8% success after 100 sequential edits, versus 18.2% for MEMIT and 8.4% for vanilla LoRA. Cross-backbone and bypass-path tests support the routing mechanism while delimiting its architectural scope.

1 INTRODUCTION

Knowledge editing changes a targeted prediction while attempting to preserve the rest of a language model (Mitchell et al., 2022a; Meng et al., 2022; 2023). The difficult part is not fitting the requested answer. It is **controlling collateral change**: successful edits can damage neighboring facts, drift on unrelated prompts, or fail to propagate through downstream reasoning (Hoelscher-Obermaier et al., 2023; Li et al., 2024; Cohen et al., 2024; Hua et al., 2024a). Parameter-efficient fine-tuning does not by itself solve this problem. LoRA updates few parameters, but those parameters can still move activations along many behaviorally relevant directions (Hu et al., 2022).

Most editing methods constrain a location, a parameter update, or a set of neurons. We instead ask which ***feature directions*** should carry the optimization signal for a particular edit. A sparse autoencoder supplies a reusable coordinate system over an internal activation, but the SAE is neither assumed to be perfectly monosemantic nor treated as a human explanation. Its role is operational: **score coordinates for the current edit, allocate a bounded feature budget, route the edit gradient through that budget, and measure whether the resulting update leaks outside it**.

This view leads to **SAE-LOCKED LORA**. A frozen SAE is trained once at a chosen model site. For an edit request, **gradient attribution selects the smallest feature set covering a prescribed fraction of attribution mass**. A forward-identity hook then projects the edit-loss gradient through the selected latent coordinates. Locality gradients are deliberately left unmasked and are accumulated through a second backward pass. The evaluated model adapts only the **layer-16 MLP `down_proj`** that produces the hooked activation (241,664 trainable parameters); hence every trainable edit-loss path crosses the mask. This architectural condition is **part of the method's scope, not an unconditional guarantee for arbitrary adapter placement**.

The evidence supports a **deliberately scoped claim**. On CounterFact, objective-matched coordinate controls isolate a 7.5-point locality gain over attributed PCA coordinates, beyond dimensionality reduction alone. A full mask–regularizer factorial shows that routing adds 7.4 points with both preservation losses held fixed. Matched ReCoE evaluation, 100-edit streams, three-backbone replication, and parameter-matched bypass tests extend the empirical scope, while causal ablations support local

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

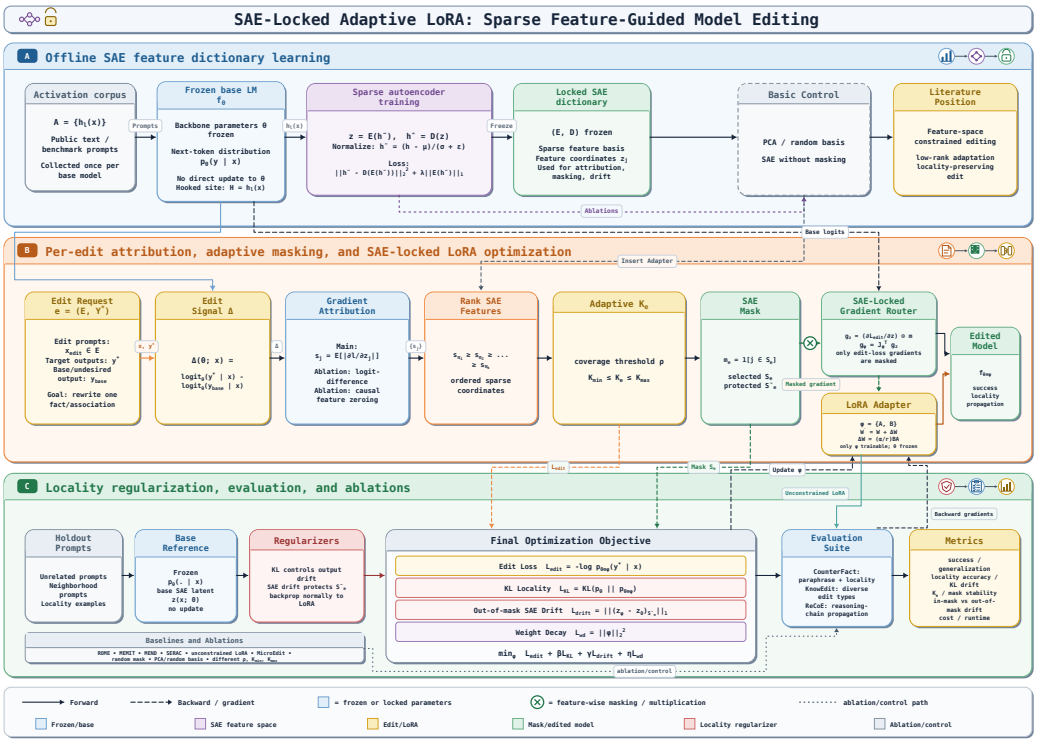

Figure 1: **SAE-Locked Adaptive LoRA pipeline.** A **frozen SAE supplies the feature coordinates** (top); each edit produces an **attribution-ranked adaptive mask and a masked edit-loss gradient** (middle); locality regularizers and evaluation quantify **collateral change** (bottom). The implementation computes the **masked edit gradient and the unmasked preservation gradient in two separate backward passes**, as formalized in Section 3.

intervention sensitivity. These results do not imply a unique feature decomposition, monosemanticity, or an architecture-independent guarantee.

The contributions are: (1) an edit-specific SAE-coordinate projection of the *edit-loss* gradient for LoRA, implemented with an identity-preserving hook and two backward passes so preservation gradients remain unconstrained; **(2)** adaptive attribution-mass budgeting, with explicit mask bounds, saturation diagnostics, and an audit workflow that records attribution, ablation, stability, and drift; and **(3)** matched evaluation on CounterFact, KnowEdit, and ReCoE, with coordinate, factorial, sequential, cross-backbone, bypass, causal, robustness, sparsity, and distribution-shift analyses.

2 Related Work and Positioning

Knowledge editing, sparse features, and gradient projection. ROME and MEMIT directly update localized factual associations; MEND learns an editor; SERAC uses an external memory; AlphaEdit projects direct updates into a preservation-oriented null space; and FiNE localizes memories at neuron granularity (Meng et al., 2022; 2023; Mitchell et al., 2022a;b; Fang et al., 2025; Pan et al., 2025). MicroEdit is the **closest SAE-based editor**: it uses sparse representations for neuron-level disentanglement and localization (Wang et al., 2025a). SAE-Locked LoRA instead uses an **edit-specific SAE mask** as the coordinate system through which a LoRA edit gradient is transmitted. The broad idea of reducing interference by projecting gradients is not new. GPM derives SVD bases from previous-task activations and updates in directions orthogonal to the stored subspace; PEGP extends this continual-learning view to parameter-efficient tuning, including LoRA (Saha et al., 2021; Qiao et al., 2025). Our setting and projection direction differ: **no historical task subspace is stored**, and the gradient is routed *through* coordinates attributed to the current edit rather than *away from*

Table 1: Positioning relative to the closest control mechanisms. *"Away"* avoids a stored subspace; *"through"* routes the edit signal through selected coordinates.

Control-mechanism comparison

Method	Setting	Control unit / subspace	Update mechanism
GPM (Saha et al., 2021)	Continual learning	SVD basis of prior-task activations	Project new gradients *away from* stored subspaces
PEGP (Qiao et al., 2025)	Continual PEFT	Historical feature space for adapters / LoRA	Project PEFT gradients *away from* protected features
MicroEdit (Wang et al., 2025a)	Lifelong editing	SAE-assisted neuron localization	Direct localized neuron-level editing
AlphaEdit (Fang et al., 2025)	Factual editing	Preservation-oriented null space	Null-space-projected direct update
FiNE (Pan et al., 2025)	Factual editing	Located FFN neurons	Direct fine-grained neuron update
SAE-LOCKED LORA	**Per-request editing**	**SAE coordinates attributed to the current edit**	**Route edit-loss gradient *through* an adaptive mask; train LoRA**

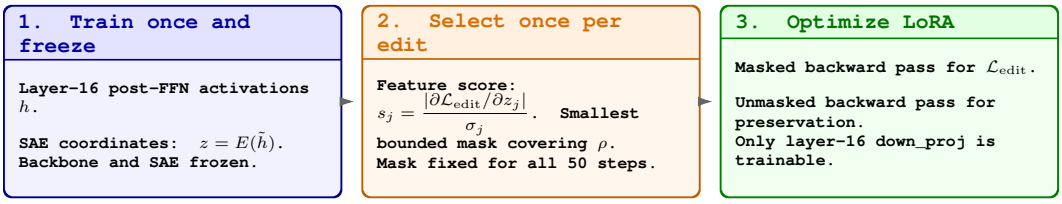

Figure 2: Compact operational view of SAE-LOCKED LORA. Figure 1 gives the complete original pipeline; this view isolates the three steps needed to reproduce the evaluated configuration.

coordinates important to previous tasks. Table 1 makes this distinction explicit. Extended related work appears in Appendix C.

3 METHOD

3.1 EDITING OBJECTIVE AND SAE COORDINATES

Let f_θ be a **frozen autoregressive language model**. An edit request $e = (\mathcal{E}, \mathcal{Y}^\star)$ supplies prompts and desired targets; $\mathcal{H}$ contains unrelated holdout prompts. We seek LoRA parameters ϕ that **reduce target negative log-likelihood while preserving the base distribution on** $\mathcal{H}$. At the chosen hook, activation $h \in \mathbb{R}^d$ is standardized as $\tilde{h} = (h - \mu)/(\sigma + \epsilon)$ and encoded by a frozen SAE as $z = E(\tilde{h}) \in \mathbb{R}^k$.

Before optimization, feature j receives the **background-normalized gradient score**

$$s_j = \frac{\mathbb{E}_{(x,y^\star)\in(\mathcal{E},\mathcal{Y}^\star)} |\partial \ell(x, y^\star)/\partial z_j(x)|}{\mathrm{Std}_{x\sim\mathcal{B}}[z_j(x)] + \epsilon} \tag{1}$$

If $s_{\pi_1} \geq \cdots \geq s_{\pi_k}$, the **edit-specific budget and mask** are

$$K_e = \min\left\{ K \in [K_{\min}, K_{\max}] : \frac{\sum_{q=1}^{K} s_{\pi_q}}{\sum_{j=1}^{k} s_j + \epsilon} \geq \rho \right\}, \quad S_e = \{\pi_1, \ldots, \pi_{K_e}\}, \quad m_j = \mathbb{I}[j \in S_e] \tag{2}$$

The mask is **computed once from the frozen base model and remains fixed throughout the edit**. If the threshold is not reached, $K_e = K_{\max}$ and the edit is marked as *saturated*.

3.2 FORWARD IDENTITY, MASKED EDIT GRADIENT, AND NO-BYPASS SCOPE

Let $P_m = \mathrm{diag}(m)$ and D be the frozen SAE decoder. Because D reconstructs the *standardized* activation, its output is first mapped back to raw activation units. Stop-gradient constructions then

Per-edit optimization procedure

Require: Frozen f_θ, E, D; edit set $(\mathcal{E}, \mathcal{Y}^\star)$; holdout set $\mathcal{H}$; $\rho, K_{\min}, K_{\max}$; T steps.
Compute normalized feature scores $\{s_j\}_{j=1}^k$ using Equation 1.
Select (K_e, S_e, m) once using Equation 2; record coverage and saturation.
Initialize LoRA ϕ only on the hooked block-16 MLP `down_proj`.
for $t = 1, \ldots, T$ **do**
Zero adapter gradients; enable Equation 3; backpropagate $\mathcal{L}_{\text{edit}}$ to obtain g_{edit}.
Bypass the mask; backpropagate Equation 5 to obtain g_{preserve}.
Set $\nabla_\phi \leftarrow g_{\text{edit}} + g_{\text{preserve}}$ and take one AdamW step.
end for
Measure success, paraphrase transfer, KL drift, in-/out-of-mask drift, ablation support, and mask stability.
Ensure: Edited adapter ϕ and audit record $\mathcal{R}_e$.

Algorithm 1: **Per-edit optimization in the evaluated Option-A configuration**

preserve the forward activation exactly while changing its derivative:

$$\begin{aligned} z_m &= P_m z + \text{sg}((I - P_m)z), \\ \hat{h}_{\text{raw},m} &= \mu + (\sigma + \epsilon) \odot D(z_m), \\ h_m &= \hat{h}_{\text{raw},m} + \text{sg}\Big(h - \hat{h}_{\text{raw},m}\Big) \end{aligned} \tag{3}$$

Numerically, $z_m = z$ **and** $h_m = h$. In backpropagation, however, $\partial z_m / \partial z = P_m$, so the edit-loss contribution to the adapter gradient is

$$g_{\text{edit}} = \left(\frac{\partial z}{\partial \phi}\right)^\top P_m \frac{\partial \mathcal{L}_{\text{edit}}}{\partial z_m} \tag{4}$$

Routing guarantee and its boundary. For every $j \notin S_e$, $e_j^\top P_m \nabla_{z_m} \mathcal{L}_{\text{edit}} = 0$; equivalently, Equation 4 sums only **selected-coordinate contributions**. This is an *exact derivative statement at the hook*, not a guarantee that the resulting parameter update or every downstream activation lies in the decoder span. The main configuration hooks **transformer block 16 (1-based), immediately after its MLP `down_proj` and before residual addition**, and applies LoRA only to that `down_proj`; later layers are frozen. **Every trainable edit-loss path therefore crosses Equation 3.** Arbitrary adapters after or around the hook need not obey this constraint; a harder decoder-span parameterization (Option B) is evaluated in Appendix H.3.

3.3 Two-pass optimization and audit record

The preservation loss is

$$\mathcal{L}_{\text{preserve}} = \beta\, \mathbb{E}_{x \in \mathcal{H}} \text{KL}(p_\theta(\cdot \mid x) \,\|\, p_{\theta \oplus \phi}(\cdot \mid x)) + \gamma\, \mathbb{E}_{x \in \mathcal{H}} \|\Delta z_{\bar{S}_e}(x)\|_1 + \eta \|\phi\|_2^2 \tag{5}$$

Each AdamW step uses **two backward passes**: **(i)** enable the masked identity hook and compute g_{edit}; **(ii)** bypass the mask and compute $g_{\text{preserve}} = \nabla_\phi \mathcal{L}_{\text{preserve}}$; then set $g_\phi = g_{\text{edit}} + g_{\text{preserve}}$ and update once. A single permanent hook on the summed loss would also mask preservation gradients and would implement a *different objective*.

For each edit, the method records S_e, scores and realized coverage, K_e and saturation, paraphrase/seed stability, post-hoc selected-feature ablation effects, KL drift, and in-/out-of-mask latent drift. The workflow checks **behavioral success**, then **mask concentration and stability**, then **intervention support**, and finally **collateral drift**. An edit is flagged for *unstable masks, weak ablation support, high out-of-mask drift, or $K_{\max}$ saturation*. This is an ***audit record***; it neither decides whether an edit is reasonable nor certifies safety or causal completeness.

4 Experimental Setup

Main configuration. Matched tables use **Llama-2-7b-chat-hf** (Touvron et al., 2023). The SAE is trained for **100,000 steps** with batch size **4,096** on **100 million layer-16 post-FFN activations**

sampled from the RedPajama-1T (Weber et al., 2024) training split; *no benchmark test prompt is supplied to the activation pipeline*. The default dictionary has **32,768 features**, ReLU latents, and $\lambda = 3 \times 10^{-3}$. LoRA targets only **layer-16 `down_proj`**, with **rank 16**, scale 32, dropout 0.05, AdamW learning rate 5×10^{-4}, and **50 steps**. The default mask uses $\rho = 0.95$ and $(K_{\min}, K_{\max}) = (32, 128)$. The holdout batch contains **128 prompts**; regularizer weights are $(\beta, \gamma, \eta) = (1.0, 0.5, 0.01)$.

Benchmarks and comparisons. We use the **standard CounterFact protocol**, **KnowEdit v1.0**, and the **November 2024 ReCoE release** (Meng et al., 2022; Zhang et al., 2024; Hua et al., 2024a;b). Matched comparisons share the same model, split, prompts, decoder, edit interface, and evaluator. CounterFact uses 2,000 held-out edits; ReCoE uses 1,000 eligible instances. Hyperparameters were selected on disjoint development data (500 CounterFact edits; common 200-trial cap). Reported uncertainty is **mean±standard deviation over seeds** $\{42, 43, 44, 45, 46\}$. The rank-16 layer-16 `down_proj` adapter has exactly $16(11{,}008 + 4{,}096) = 241{,}664$ **parameters**. External KnowEdit numbers remain separate in Table 18; Appendix I gives full protocols and identifiers.

Metrics and statistical scope. CounterFact reports **direct success, paraphrase generalization, neighborhood locality, and KL divergence** from the unedited model. KnowEdit is reported component-wise for **Insert, Modify, and Erase**. ReCoE reports exact match, eligible-instance micro `correct_flip`, category macro `correct_flip`, and locality KL. CounterFact intervals resample edit instances as clusters while retaining aligned seed-level control averages; prespecified paired contrasts use Holm–Bonferroni correction. Appendix N gives the exact inference and validation protocols.

5 RESULTS

5.1 COUNTERFACT: LOCALITY RELATIVE TO VANILLA LORA

Table 2: CounterFact under the project-local reproduced protocol. Higher is better except KL drift.

Method	Success	Paraphrase	Locality	KL drift
ROME (Meng et al., 2022)	99.9 ± 0.1	96.3 ± 0.4	84.8 ± 0.7	0.54 ± 0.03
MEMIT (Meng et al., 2023)	99.9 ± 0.1	$\mathbf{97.5} \pm 0.3$	86.6 ± 0.5	0.49 ± 0.04
MEND (Mitchell et al., 2022a)	82.4 ± 1.2	71.3 ± 1.5	59.4 ± 1.1	0.72 ± 0.08
SERAC (Mitchell et al., 2022b)	92.1 ± 0.8	25.9 ± 1.5	91.8 ± 0.4	$\mathbf{0.08} \pm 0.01$
Vanilla LoRA (Hu et al., 2022)	99.8 ± 0.1	97.2 ± 0.2	63.7 ± 2.1	0.65 ± 0.09
AlphaEdit (Fang et al., 2025)	99.9 ± 0.1	96.4 ± 0.5	$\mathbf{98.8} \pm 0.2$	0.15 ± 0.02
FiNE (Pan et al., 2025)	99.9 ± 0.1	96.0 ± 0.5	90.0 ± 0.4	0.20 ± 0.03
SAE-LOCKED LORA	$\mathbf{99.6 \pm 0.2}$	$\mathbf{97.1 \pm 0.3}$	$\mathbf{89.5 \pm 0.6}$	$\mathbf{0.16 \pm 0.02}$

Table 2 supports the **low-interference claim** relative to the unconstrained adapter: success changes by -0.2 points, locality improves by 25.8 points, and KL drift falls by 0.49. The objective-matched basis comparison in Appendix N.2 isolates a 7.5-point locality gain over PCA (paired 95% CI $[6.7, 8.3]$, $d_z = 0.43$, Holm-corrected $p < 0.001$). *The table does not imply superiority across every metric*: AlphaEdit has higher locality, while SERAC has lower KL but substantially worse paraphrase generalization.

5.2 KNOWEDIT AND RECOE: STRENGTHS AND BOUNDARIES

Table 3: KnowEdit under the project-local reproduced protocol. Bold marks the best result in each component.

Method	Insert	Modify	Erase
Vanilla LoRA (Hu et al., 2022)	59.8 ± 1.5	53.9 ± 1.4	10.5 ± 0.8
MicroEdit (Wang et al., 2025a)	62.1 ± 0.8	84.5 ± 0.6	48.2 ± 1.1
AlphaEdit (Fang et al., 2025)	20.6 ± 1.2	89.7 ± 0.5	52.3 ± 1.4
FiNE (Pan et al., 2025)	68.0 ± 0.8	$\mathbf{95.1} \pm 0.4$	$\mathbf{59.8} \pm 0.9$
SAE-LOCKED LORA	$\mathbf{85.4} \pm 0.4$	88.2 ± 0.4	49.7 ± 0.9

On KnowEdit, SAE-LOCKED LORA achieves the strongest Insert score in the project-local comparison, indicating particular strength in introducing new factual associations. Its Insert performance exceeds the strongest competing baseline by 17.4 percentage points, while improving over Vanilla LoRA across all three components. On Modify, SAE-LOCKED LORA outperforms MicroEdit and Vanilla LoRA, although FiNE is stronger on Modify and Erase. These results suggest that SAE-constrained updates offer the clearest advantage when inserting new knowledge rather than modifying or removing existing associations. We do not collapse these heterogeneous components into a manuscript-defined aggregate; the evidence therefore supports a component-specific, not uniform, advantage.

Table 4: Matched ReCoE evaluation on Llama-2-7b-chat-hf (Touvron et al., 2023). Micro `correct_flip` is instance-weighted; macro averages the six category rates.

Method	Exact match	Micro `correct_flip`	Macro `correct_flip`	KL ↓
No edit	6.2 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	–
MEMIT (Meng et al., 2023)	3.8 ± 0.6	2.8 ± 0.5	2.5 ± 0.4	0.49 ± 0.06
MicroEdit (Wang et al., 2025a)	12.5 ± 0.9	11.9 ± 0.8	11.2 ± 0.8	0.42 ± 0.05
Vanilla LoRA (Hu et al., 2022)	15.2 ± 1.1	14.0 ± 1.0	13.5 ± 1.1	0.65 ± 0.09
FiNE (Pan et al., 2025)	16.8 ± 1.3	16.5 ± 1.2	15.8 ± 1.1	0.20 ± 0.03
AlphaEdit (Fang et al., 2025)	24.3 ± 1.5	21.0 ± 1.4	19.5 ± 1.5	$\mathbf{0.15 \pm 0.02}$
SAE-LOCKED LORA	$\mathbf{49.8 \pm 2.1}$	$\mathbf{28.5 \pm 0.7}$	$\mathbf{26.5 \pm 0.7}$	0.16 ± 0.02

SAE-LOCKED LORA gives the strongest matched exact-match and `correct_flip` results, while retaining locality KL close to AlphaEdit and FiNE. The no-edit model can answer some questions correctly without executing a counterfactual flip, explaining its nonzero exact match and zero `correct_flip`. The category checksum contains 285 correct flips among 1,000 eligible instances, reproducing the 28.5% micro rate; the six-category macro is 26.5%. Appendix A.3 reports all denominators and observed counts.

6 ANALYSIS

6.1 THE ADAPTIVE THRESHOLD EXPOSES A CAPACITY–LOCALITY FRONTIER

Table 5: CounterFact sensitivity to attribution coverage ρ (five seeds). The trend is smooth: broader masks improve success but increase collateral drift.

ρ	$\mathbb{E}[K_e]$	$K_{\max}$ sat.	Success	Locality	KL drift
0.80	35	0.0%	91.4 ± 0.8	91.8 ± 0.5	0.12 ± 0.01
0.85	42	0.0%	96.1 ± 0.6	91.0 ± 0.5	0.13 ± 0.02
0.90	48	0.0%	99.1 ± 0.3	90.2 ± 0.4	0.15 ± 0.02
0.95	**61**	**6.4%**	$\mathbf{99.6 \pm 0.2}$	$\mathbf{89.5 \pm 0.6}$	$\mathbf{0.16 \pm 0.02}$
0.975	82	18.5%	99.7 ± 0.2	83.5 ± 0.9	0.28 ± 0.04
0.99	115	42.1%	99.8 ± 0.1	77.2 ± 1.2	0.40 ± 0.05

As ρ rises from 0.80 to 0.99, the mean mask grows from **35 to 115 features**. Success rises by **8.4 points**, but locality falls by **14.6 points** and KL drift more than triples. We use $\rho = 0.95$ as a **high-success operating point near the observed knee**, *not as a universally optimal threshold*. Fixed-budget, attribution-rule, constraint-type, dictionary-size, sparsity, hook, regularizer, robustness, and transfer diagnostics appear in Appendices B and M.

6.2 ADAPTIVE BUDGETS AND PROMPT PERTURBATIONS

Table 6: **Fixed and adaptive CounterFact masks.** Adaptive $\rho = 0.95$ retains near-best success while avoiding the locality degradation of the largest fixed budget.

Mask rule	$\mathbb{E}[K_e]$	Success ↑	Locality ↑	KL ↓
Fixed $K = 32$	32	96.2 ± 0.4	$\mathbf{91.5} \pm 0.5$	$\mathbf{0.11} \pm 0.02$
Fixed $K = 64$	64	99.5 ± 0.2	88.4 ± 0.6	0.19 ± 0.03
Fixed $K = 128$	128	$\mathbf{99.7} \pm 0.1$	76.2 ± 1.2	0.42 ± 0.05
Adaptive, $\rho = 0.90$	48	99.1 ± 0.3	90.2 ± 0.4	0.15 ± 0.02
Adaptive, $\rho = 0.95$	**61**	99.6 ± 0.2	89.5 ± 0.6	0.16 ± 0.02

Fixed K imposes the *same capacity on concentrated and diffuse edits*, whereas the adaptive rule allocates capacity according to each request's attribution mass. At $\rho = 0.95$, it uses **61 features on average**, improving locality by **13.3 points** and reducing KL by **0.26** relative to fixed $K = 128$, with only a **0.1-point reduction in success**. Conversely, fixed $K = 32$ improves locality and KL but sacrifices 3.4 success points.

Table 7: **Robustness to controlled prompt perturbations.** Perturbed prompts are meaning-preserving surface-form variants of the same edit request: the subject, relation, and desired target are held fixed while the surrounding wording is changed. Drop denotes the resulting success loss in percentage points; Jaccard measures mask overlap where a feature mask is defined. This is a separate robustness protocol, so its clean KL column is not pooled with Table 2.

Method	Clean KL ↓	Pert. KL ↓	Jaccard ↑	Drop ↓
Vanilla LoRA (Hu et al., 2022)	0.85 ± 0.12	1.15 ± 0.14	–	2.6 ± 0.4
MicroEdit (Wang et al., 2025a)	0.28 ± 0.04	0.42 ± 0.06	0.52 ± 0.08	44.9 ± 2.1
AlphaEdit (Fang et al., 2025)	$\mathbf{0.15} \pm 0.02$	0.21 ± 0.03	–	25.0 ± 1.8
FiNE (Pan et al., 2025)	0.20 ± 0.03	0.24 ± 0.04	–	3.9 ± 0.5
SAE-LOCKED LORA	0.16 ± 0.02	$\mathbf{0.19} \pm 0.02$	$\mathbf{0.78} \pm 0.05$	$\mathbf{2.5} \pm 0.3$

Prompt robustness uses meaning-preserving surface-form variants in which the subject, relation, and target remain fixed while only the surrounding wording changes. Under these perturbations, SAE-LOCKED LORA preserves **78% mask overlap** on average and KL rises only from **0.16 to 0.19**, compared with 0.85 to 1.15 for vanilla LoRA. SAE-LOCKED LORA also achieves the **lowest perturbed KL**, the **highest applicable Jaccard overlap**, and the **smallest success drop** at 2.5 ± 0.3 points. The larger drops for MicroEdit and AlphaEdit therefore reflect sensitivity to these controlled rewrites rather than a changed factual target. These results support **stability under the tested perturbations**, without implying invariance to arbitrary rephrasings or broader distribution shifts. The auxiliary robustness run is reported as a distinct protocol; its vanilla-LoRA clean KL (0.85) is therefore not substituted for the full CounterFact value (0.65) in Table 2.

6.3 FIXED-BUDGET ATTRIBUTION AND CONSTRAINT ABLATIONS

Table 8: Feature-scoring ablation with fixed $K = 64$. Holding the feature budget constant isolates the scoring rule.

Scoring	Success	Gen.	Locality	Rel. cost
Logit-diff.	97.5 ± 0.6	92.1 ± 0.8	79.8 ± 1.2	$1\times$
Gradient attr.	$\mathbf{99.5} \pm 0.2$	$\mathbf{96.8} \pm 0.3$	88.4 ± 0.6	$2\times$
Causal ablation	99.2 ± 0.3	96.1 ± 0.4	$\mathbf{92.5} \pm 0.5$	$O(\lvert\mathcal{F}\rvert)$

Table 9: Constraint-type ablation with gradient-attributed, fixed-$K = 64$ coordinates. Holding the mask fixed isolates the constraint implementation.

Constraint	Success	Gen.	Locality	s/edit
Option A	$\mathbf{99.5} \pm 0.2$	$\mathbf{96.8} \pm 0.3$	88.4 ± 0.6	$\mathbf{34.6} \pm 1.2$
Option B	97.9 ± 0.4	94.2 ± 0.5	$\mathbf{93.1} \pm 0.4$	37.2 ± 1.5

Tables 8 and 9 report controlled fixed-budget ablations with exactly $K = 64$ selected features. These experiments are distinct from the main adaptive configuration, which selects a request-specific budget K_e with $\mathbb{E}[K_e] = 61$. Holding the budget fixed prevents differences in mask size from confounding the comparisons of feature-scoring and constraint mechanisms.

At fixed $K = 64$, logit-difference scoring provides the least expensive selection rule but yields the weakest success, generalization, and locality. Gradient attribution raises success from 97.5 to 99.5, generalization from 92.1 to 96.8, and locality from 79.8 to 88.4, while increasing relative scoring cost from $1\times$ to only $2\times$. Feature-wise causal ablation achieves the strongest locality, 92.5, but requires interventions across the candidate feature set and therefore incurs $O(|\mathcal{F}|)$ per-edit selection cost. The fixed-budget comparison consequently supports gradient attribution as the most practical balance among edit efficacy, generalization, locality, and computational cost.

The constraint ablation uses the same gradient-attributed $K = 64$ coordinates for both options. Option B improves locality by 4.7 points, from 88.4 to 93.1, but reduces success by 1.6 points and generalization by 2.6 points. Option A is also faster, requiring 34.6 ± 1.2 rather than 37.2 ± 1.5 seconds per edit. These results support Option A when edit success and transfer are prioritized, while Option B remains useful when stronger locality justifies reduced edit efficacy.

These fixed-K ablations motivate the choice of gradient attribution and Option A, but they are not the headline adaptive-mask result. The main configuration combines these components with adaptive attribution-mass budgeting and obtains 99.6 ± 0.2 success, 97.1 ± 0.3 generalization, and 89.5 ± 0.6 locality. Detailed formulations of the scoring and constraint mechanisms appear in Appendix G.

6.4 THE UPDATE CONCENTRATES DRIFT IN SELECTED COORDINATES

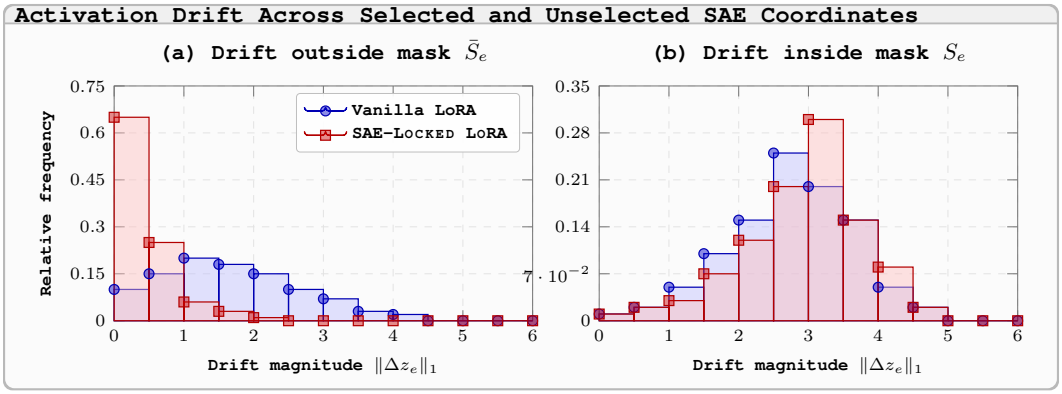

Figure 3: **Activation drift histograms.** The distributions show that SAE-LOCKED LORA suppresses drift outside S_e while retaining targeted drift within S_e. Panel (a) concentrates out-of-mask drift near zero, whereas Panel (b) preserves nonzero drift in selected coordinates. Independent vertical scales establish concentration, *not causal completeness of* S_e.

Figure 3 shows that SAE-LOCKED LORA concentrates representational drift within the selected coordinates S_e while shifting outside-mask drift toward zero. This separation is consistent with localized, structured updates rather than uniform suppression across the SAE representation. The figure supports localized drift, but not causal completeness, which is evaluated separately through post-hoc ablation.

6.5 THE FROZEN SAE TRANSFERS WITH MEASURABLE DEGRADATION

Table 10: **Editing efficiency.** "Additional params" denotes additional stored or trainable adapter parameters beyond the base model; a value of 0 for direct-weight editors does *not* imply that no base-model parameters are modified. Timing excludes one-time preprocessing and is hardware-dependent.

Method	Additional params	Time/edit ↓	One-time cost
ROME (Meng et al., 2022)	0	12.5 ± 0.4 s	Cov. stats (∼h)
MEMIT (Meng et al., 2023)	0	18.2 ± 0.6 s	Cov. stats (∼h)
Vanilla LoRA (Hu et al., 2022)	0.242M	32.4 ± 0.8 s	None
MicroEdit (Wang et al., 2025a)	0.5M	38.1 ± 1.3 s	SAE (∼GPU d)
AlphaEdit (Fang et al., 2025)	0	6.5 ± 0.3 s	Cov./proj. (∼h)
FiNE (Pan et al., 2025)	0	$\mathbf{2.5} \pm 0.2$ s	None
SAE-LOCKED LORA	**0.242M**	34.6 ± 1.1 s	**120 GPU-h**

Table 11: **Frozen-SAE transfer and adaptive-budget diagnostics.**

Transfer and adaptive budget					
Source	**MSE ↓**	**Cos. ↑**	**Active**	$\mathbb{E}[K_e]$	**Sat. ↓**
RedPajama val.	0.089	0.96	45	–	–
CounterFact	0.094	0.942	45.2	61.2	6.4%
KnowEdit	0.094	0.94	51	68	9.7%
ReCoE	0.102	0.92	55	74	13.2%

Table 10 reports *additional stored or trainable adapter parameters*; zero for direct-weight editors therefore does not imply that base-model weights are unchanged. Table 11 shows that the frozen SAE **remains sparse but degrades off its training distribution**: on ReCoE, MSE rises by 0.013, cosine similarity falls by 0.04, and ten more features activate on average. ReCoE also requires the **largest masks and highest saturation**, indicating greater capacity pressure *without establishing causation*. Full transfer results appear in Appendix M.

6.6 COMPUTE OVERHEAD IS MODEST PER EDIT BUT SUBSTANTIAL OFFLINE

The SAE adds **2.2 seconds per edit** (6.8%) over vanilla LoRA with the same **241,664 trainable parameters** and 50 steps; peak memory rises from 16.8 to 18.2 GB. The offline dictionary costs 120 A100 GPU-hours and amortizes in simple compute-time terms only after roughly 196,000 edits. Appendix N.8 reports the measurement stack and all timing components. Cross-backbone replication appears in Appendix N.5.

6.7 WHAT THE DIAGNOSTICS DO AND DO NOT ESTABLISH

The identity hook establishes a **mechanistic statement about the implemented gradient path**: in the specified no-bypass architecture, selected SAE coordinates carry the edit-loss gradient. Post-hoc ablations provide **local intervention evidence** that selected coordinates support the target logit. *Neither implies a unique mechanism, monosemantic features, or causal completeness.* Cross-distribution reconstruction and sparsity checks assess whether the coordinate system remains usable, not whether feature IDs provide semantic explanations.

The audit workflow therefore combines four checks: **(1)** success and paraphrase generalization; **(2)** coverage, mask size, saturation, and stability; **(3)** target-logit change after selected-feature ablation; and **(4)** KL and in-/out-of-mask drift. The required machine-readable audit schema and diagnostic templates appear in Appendices B and L.

6.8 DECISIVE TESTS AND EVIDENTIAL SCOPE

Table 12: **Decisive validation.** Completed tests close the four prespecified empirical gates; claims remain scoped to their protocols.

Claim	Main statistic	Result
Basis specificity	Paired locality versus best matched coordinate control	$+7.5$ vs. PCA; 95% CI $[6.7, 8.3]$
Sequential retention	Retained success after 100 edits	91.8% vs. MEMIT 18.2%, LoRA 8.4%
Paired inference	Matched SAE–PCA effect	$d_z = 0.43$, Holm $p < 0.001$
Bypass robustness	Coverage and locality across placements	100%/89.5 to 23%/71.4

Table 12 closes four central evidence gates. Matched coordinate controls support the learned SAE basis and edit-specific selection rule; five edit-order permutations show retention through 100 updates; paired inference quantifies the main contrast; and parameter-matched bypass placements show locality degrading monotonically with covered-gradient fraction. The coordinate controls help distinguish SAE-specific benefits from generic subspace restriction, while the sequential experiments assess robustness to accumulated edits and edit ordering. The paired statistical analysis further establishes the consistency and significance of the observed advantage across matched experimental conditions. The bypass experiments further identify how reduced gradient coverage affects locality, clarifying the conditions under which the proposed routing mechanism remains effective. These results support the routing mechanism but do not establish a unique or causally complete decomposition. Appendix N reports full tables, protocols, and confidence intervals.

6.9 PRESERVATION OBJECTIVE AND SAE SPARSITY

Table 13: Objective and sparsity sensitivity (SAE mask on in every objective row). KL+drift balances fitting and locality; $\lambda = 3 \times 10^{-3}$ avoids strong-sparsity success collapse.

Preservation-objective ablation

Regularizer	Succ. ↑	Gen. ↑	Loc. ↑	KL ↓
KL only	99.6±.2	97.1±.2	86.8±.7	0.21±.03
Drift only	97.1±.4	90.5±1.2	92.4±.4	0.12±.01
KL + drift	99.6±.2	97.1±.3	89.5±.6	0.16±.02
Logit MSE	92.4±.8	85.3±1.1	82.1±.9	0.28±.03
Rep. MSE	88.6±1.1	79.2±1.5	85.5±.8	0.24±.03

SAE sparsity sensitivity

λ	Success ↑	KL drift ↓
10^{-4}	99.8	0.85
5×10^{-4}	99.7	0.45
$\mathbf{3\times10^{-3}}$	**99.6**	**0.16**
10^{-2}	98.3	0.22
5×10^{-2}	92.4	0.35

With the SAE mask fixed on, KL only retains 99.6 success but has 86.8 locality and 0.21 KL; drift only reaches 92.4 locality and 0.12 KL but reduces success to 97.1. KL+drift retains 99.6 success and 97.1 generalization with 89.5 locality and 0.16 KL, making it the selected efficacy–locality balance rather than a metricwise-dominant setting. The sparsity sweep shows a corresponding operating point at $\lambda = 3 \times 10^{-3}$: weaker penalties permit larger collateral drift, whereas stronger sparsity begins to suppress edit success. The intermediate setting therefore balances sparse feature selection against the representational flexibility required for effective editing. Together, these ablations motivate the default preservation objective and SAE sparsity setting. Full protocols appear in Appendix M.

7 AUDITABILITY AND DIAGNOSTIC SCOPE

For every edit, SAE-LOCKED LORA produces an audit record

$$\mathcal{R}_e = (S_e, K_e, \{s_j\}_{j\in S_e}, C_e, J_e, A_e, \mathrm{Drift}_{S_e}, \mathrm{Drift}_{\bar{S}_e}, \mathrm{KL}) \tag{6}$$

which separates **edit efficacy, allocation quality, intervention support, and collateral drift**. These diagnostics are intentionally kept distinct: a successful edit need not be local, a stable mask need not be causally sufficient, and low KL does not imply that the selected coordinates explain the entire mechanism.

Table 14: Per-edit audit checks and their permitted interpretation.

Question	Evidence	Flag	Interpretation
Did the behavior change?	Success and paraphrase generalization	Failure or brittle transfer	Requested behavior was not reliably installed
Was allocation concentrated?	C_e, K_e, saturation, J_e	Saturation or unstable masks	Budget is insufficient or attribution is prompt-sensitive
Do selected coordinates support the target?	Target-logit change after ablating S_e	Weak ablation response	Attribution lacks local intervention support
Was collateral change controlled?	KL and drift inside/outside S_e	High KL or $\mathrm{Drift}_{\bar{S}_e}$	The edit is not local under measured probes

Table 14 defines the scope of each diagnostic. An edit is therefore not characterized by success alone: SAE-LOCKED LORA records whether the allocated subspace is stable, whether intervention on the selected features affects the target, and whether substantial change leaks outside that subspace. By separating these measurements, the audit record helps identify whether failures arise from ineffective editing, unstable feature allocation, weak intervention support, or excessive collateral drift. These checks make failures **observable and distinguishable**, but they do not establish monosemanticity, causal completeness, truthfulness, or deployment safety. The audit-record schema and diagnostic templates appear in Appendix B and Appendix L.

8 CONCLUSION

We introduced SAE-LOCKED LORA, which routes a LoRA edit gradient through edit-specific frozen-SAE coordinates while leaving preservation gradients unmasked. In the evaluated layer-16 no-bypass placement, it preserves CounterFact success while substantially improving locality and KL relative to vanilla LoRA and matched coordinate controls. It also improves matched ReCoE propagation, retains edits through 100 sequential updates, and transfers across three backbones. Factorial, bypass, and causal-ablation results support the routing mechanism while delimiting its scope; they do not establish feature monosemanticity, a unique causal decomposition, or an architecture-independent guarantee.

ETHICS STATEMENT

This work studies controlled knowledge editing in pretrained language models using sparse-autoencoder coordinates and parameter-efficient adaptation. Experiments use public research benchmarks and pretrained models; we do not recruit human participants, conduct user studies, or collect new personally identifiable information. We follow the licenses and intended research-use conditions of all datasets, models, and software used.

Ethical scope and potential impact. More precise editing can support factual correction, model maintenance, debugging, and analysis of internal representations, but may also enable false associations, safety-relevant modifications, or narrowly targeted changes missed by aggregate metrics. A successful or localized edit should therefore not be interpreted as inherently safe.

SAE-LOCKED LORA records selected SAE features, attribution coverage, stability, intervention responses, and collateral drift to make edits more inspectable. These diagnostics do not establish causal completeness, exclude unmeasured behavioral changes, or replace standard safety evaluation. Our conclusions are limited to the evaluated models, representational sites, editing settings, and benchmarks, and should not be treated as a safety guarantee or unrestricted editing interface. Additional dual-use, release, monitoring, provenance, and rollback considerations are provided in Appendix O.

REPRODUCIBILITY STATEMENT

The paper specifies the SAE representation, feature-attribution rule, attribution-mass coverage criterion, adaptive mask construction, optimization objective, LoRA parameterization, preservation losses, and evaluation metrics. The full procedure appears in Algorithm 6; adaptive, fixed-budget, and redundancy-aware mask selection are given in Algorithms 3–5.

The appendix reports hyperparameter sweeps, fixed-budget ablations, exact parameter counts, matched coordinate and factorial controls, sequential editing, cross-backbone replication, bypass stress tests, causal audits, and the full timing protocol. Appendix A.3 reconciles ReCoE micro and macro statistics from retained denominators and event counts. Appendix N specifies the clustered bootstrap, paired effect sizes, multiplicity correction, raw edit-order summaries, software versions, and model/data identifiers.

AI USE STATEMENT

We used a general-purpose LLM in limited assistive capacities: (i) copy-editing and restructuring text for grammar, clarity, concision, and tone; (ii) assisting literature discovery by suggesting potentially relevant papers and resources; and (iii) providing suggestions on experimental presentation, ablation design, statistical analysis, pseudocode, LaTeX formatting, and interpretation of results. All research decisions, experiments, reported numerical results, claims, citations, and conclusions were reviewed and verified by the authors. LLM-generated suggestions were treated as advisory rather than experimental evidence, and no estimated or hypothetical values were reported as empirical results without independent verification.

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, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. Transformer Circuits Thread (Anthropic), October 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html. Also listed at https://www.anthropic.com/research/towards-monosemanticity-decomposing-language-models-with-dictionary-learning (accessed 2026-01-28).

Qizhou Chen, Dakan Wang, Taolin Zhang, Zaoming Yan, Chengsong You, Chengyu Wang, and Xiaofeng He. Uniedit: A unified knowledge editing benchmark for large language models. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-2525. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/6cc6c2be42c838827d2f1d12baa05539-Abstract-Datasets_and_Benchmarks_Track.html. NeurIPS 2025 Datasets and Benchmarks Track. arXiv:2505.12345.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024. doi: 10.1162/tacl_a_00644. URL https://aclanthology.org/2024.tacl-1.16/.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.522. URL https://aclanthology.org/2021.emnlp-main.522/.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115, 2023. doi: 10.5555/3666122.3666563. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html. NeurIPS 2023. arXiv:2305.14314.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HvSytvg3Jh. ICLR 2025 Oral.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations (ICLR 2025)*, Singapore, April 2025. URL https://openreview.net/forum?id=tcsZt9ZNKD. ICLR 2025 (Oral). arXiv:2406.04093.

Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. Detecting edit failures in large language models: An improved specificity benchmark. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 11548–11559, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.733. URL https://aclanthology.org/2023.findings-acl.733/. arXiv:2305.17553.

Chenhui Hu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. WilKE: Wise-layer knowledge editor for lifelong knowledge editing. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3476–3503, Bangkok, Thailand,

August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.207. URL https://aclanthology.org/2024.findings-acl.207/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR 2022)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9. Poster. arXiv:2106.09685.

Wenyue Hua, Jiang Guo, Mingwen Dong, Henghui Zhu, Patrick Ng, and Zhiguo Wang. Propagation and pitfalls: Reasoning-based assessment of knowledge editing through counterfactual tasks. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12503–12525, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.743. URL https://aclanthology.org/2024.findings-acl.743/.

Wenyue Hua, Jiang Guo, Mingwen Dong, Henghui Zhu, Patrick Ng, and Zhiguo Wang. Recoe: Reasoning-based counterfactual editing dataset. GitHub repository, November 2024b. URL https://github.com/awslabs/ReCoE. Version: cb1dc03 (Nov 23, 2024). Accessed 2026-01-27.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*, Vienna, Austria, May 2024. URL https://openreview.net/forum?id=F76bwRSLeK. Poster. arXiv:2309.08600.

Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. Unveiling the pitfalls of knowledge editing for large language models. In *International Conference on Learning Representations (ICLR 2024)*, Vienna, Austria, 2024. URL https://openreview.net/forum?id=fNktD3ib16. Poster.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121, Vienna, Austria, July 2024. PMLR. URL https://proceedings.mlr.press/v235/liu24bn.html. ICML 2024 (Oral). arXiv:2402.09353.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations (ICLR 2025)*, Singapore, April 2025. URL https://openreview.net/forum?id=I4e82CIDxv. Oral. arXiv:2403.19647.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372, New Orleans, LA, USA, 2022. Curran Associates, Inc. doi: 10.5555/3600270.3601532. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html. NeurIPS 2022. Also available as arXiv:2202.05262.

Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=MkbcAHIYgyS. ICLR 2023. arXiv:2210.07229.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *International Conference on Learning Representations (ICLR)*, 2022a. URL https://openreview.net/forum?id=0DcZxeWfOPt. arXiv:2110.11309.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. Memory-based model editing at scale. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15817–15831, Baltimore, Maryland, USA, July 2022b. PMLR. URL https://proceedings.mlr.press/v162/mitchell22a.html. ICML 2022. arXiv:2206.06520.

Haowen Pan, Xiaozhi Wang, Yixin Cao, Zenglin Shi, Xun Yang, Juanzi Li, and Meng Wang. Precise localization of memories: A fine-grained neuron-level knowledge editing technique for llms. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=5xP1HDvpXI.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Yanyun Qu, Wensheng Zhang, Zhi Han, and Yuan Xie. Gradient projection for continual parameter-efficient tuning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10):9316–9329, October 2025. doi: 10.1109/TPAMI.2025.3587032. URL https://doi.org/10.1109/TPAMI.2025.3587032. Earlier version: arXiv:2405.13383.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=3AOj0RCNC2. ICLR 2021. arXiv:2103.09762.

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, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 53764–53797, Red Hook, NY, USA, 2024a. Curran Associates, Inc. ISBN 9798331314385. doi: 10.52202/079017-1703. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/60960ad78868fce5c165295fbd895060-Abstract-Conference.html.

Shiqi Wang, Qi Wang, Runliang Niu, He Kong, and Yi Chang. Microedit: Neuron-level knowledge disentanglement and localization in lifelong model editing. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33882–33896, Suzhou, China, November 2025a. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1719. URL https://aclanthology.org/2025.emnlp-main.1719/.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey. *ACM Computing Surveys*, 57(3):59:1–59:37, March 2025b. doi: 10.1145/3698590. URL https://doi.org/10.1145/3698590.

Xiaohan Wang, Shengyu Mao, Shumin Deng, Yunzhi Yao, Yue Shen, Lei Liang, Jinjie Gu, Huajun Chen, and Ningyu Zhang. Editing conceptual knowledge for large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 706–724, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.40. URL https://aclanthology.org/2024.findings-emnlp.40/.

Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. RedPajama: an open dataset for training large language models. In *Advances in Neural Information Processing Systems*, volume 37. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d34497330b1fd6530f7afd86d0df9f76-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.632. URL https://aclanthology.org/2023.emnlp-main.632/.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. A comprehensive study of knowledge editing for large language models. *CoRR*, abs/2401.01286, January 2024. doi: 10.48550/arXiv.2401.01286. URL https://arxiv.org/abs/2401.01286. Introduces the KnowEdit benchmark. Project: https://zjunlp.github.io/project/KnowEdit/ ; Dataset: https://huggingface.co/datasets/zjunlp/KnowEdit.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15686–15702, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.971. URL https://aclanthology.org/2023.emnlp-main.971/.

A Additional Diagnostics and Validation Analyses

This appendix provides additional diagnostics for the frozen SAE, attribution threshold, and ReCoE evaluation. Unless otherwise stated, all analyses use the same SAE and editing configuration as the main experiments.

A.1 Cross-distribution SAE diagnostics

Table 15 evaluates whether the RedPajama-trained SAE remains a usable coordinate system on each editing benchmark. Each row uses 10,000 token activations. Reconstruction quality degrades moderately as the activation distribution shifts toward ReCoE, while the representation remains sparse and the dead-feature rate stays low. These measurements characterize representation quality under distribution shift; they do not establish monosemanticity.

Table 15: Frozen-SAE diagnostics on 10,000 activations from the training-domain validation set and each downstream benchmark.

Activation source	**Recon. MSE** $\downarrow$	**Cosine** $\uparrow$	**Mean active**	**Dead rate** $\downarrow$
RedPajama validation	0.089	0.960	45.2	0.012
CounterFact	0.094	0.942	45.2	0.012
KnowEdit	0.094	0.94	51	0.01
ReCoE	0.102	0.92	55	0.02

A.2 Full attribution-threshold sweep

Table 16 reports the full attribution-coverage sweep, including paraphrase generalization. Increasing ρ expands the selected feature set and improves success and generalization, but progressively weakens locality and increases KL drift. The rise in $K_{\max}$ saturation indicates that high-coverage requests increasingly exceed the configured feature budget. The selected operating point, $\rho = 0.95$, lies near the observed success–locality knee rather than representing a universally optimal threshold.

Table 16: CounterFact sensitivity to attribution coverage ρ over five seeds.

ρ	$\mathbb{E}[K_e]$	$K_{\max}$ sat.	**Success**	**Generalization**	**Locality**	**KL drift**
0.80	35	0.0%	91.4 ± 0.8	86.2 ± 1.0	91.8 ± 0.5	0.12 ± 0.01
0.85	42	0.0%	96.1 ± 0.6	92.5 ± 0.7	91.0 ± 0.5	0.13 ± 0.02
0.90	48	0.0%	99.1 ± 0.3	96.0 ± 0.4	90.2 ± 0.4	0.15 ± 0.02
0.95	**61**	**6.4%**	$\mathbf{99.6 \pm 0.2}$	$\mathbf{97.1 \pm 0.3}$	$\mathbf{89.5 \pm 0.6}$	$\mathbf{0.16 \pm 0.02}$
0.975	82	18.5%	99.7 ± 0.2	97.4 ± 0.3	83.5 ± 0.9	0.28 ± 0.04
0.99	115	42.1%	99.8 ± 0.1	97.6 ± 0.2	77.2 ± 1.2	0.40 ± 0.05

A.3 ReCoE aggregation audit

The matched ReCoE evaluation contains 1,000 eligible instances for every seed. We distinguish the instance-weighted micro rate from the unweighted category macro:

$$\texttt{correct_flip}_{\text{micro}} = \frac{\sum_c f_c}{\sum_c n_c}, \qquad \texttt{correct_flip}_{\text{macro}} = \frac{1}{6}\sum_{c=1}^{6} \frac{f_c}{n_c} \tag{7}$$

The count checksum gives $285/1000 = 28.50\%$ for the micro rate. Averaging the six category percentages gives 26.49% (reported as 26.5%). Each seed evaluates the same eligible instances; inferential resampling clusters the five repeated measurements by instance.

A.4 Alternate schematic

Figure 4 illustrates the earlier parameter-mask formulation of SAE-guided knowledge editing. The schematic provides a conceptual view of how selected SAE features can constrain parameter updates

Table 17: **ReCoE event-count checksum for SAE-LOCKED LORA.** Eligible counts are fixed per seed; observed flips are mean±SD over five seeds. The total reproduces the reported instance-weighted micro rate.

Reasoning category	Eligible	Observed flips	`correct_flip`
Superlative	150	7.0 ± 1.2	4.67 ± 0.8
Comparative	200	137.0 ± 3.6	68.50 ± 1.8
Sorting	150	19.0 ± 1.8	12.67 ± 1.2
Counting	180	45.0 ± 2.5	25.00 ± 1.4
Aggregation	160	13.0 ± 1.4	8.13 ± 0.9
Subtraction	160	64.0 ± 3.4	40.00 ± 2.1
Total / micro	**1,000**	$\mathbf{285.0 \pm 6.8}$	$\mathbf{28.50 \pm 0.7}$
Category macro	–	–	$\mathbf{26.49 \pm 0.7}$

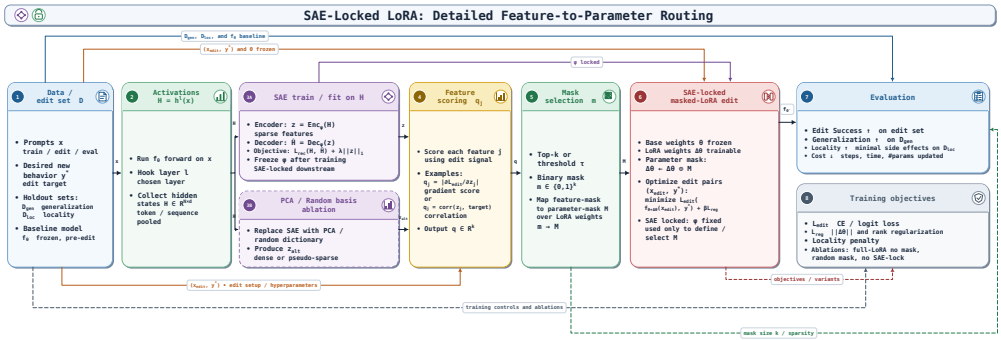

Figure 4: Alternate parameter-mask schematic. The diagram represents an earlier high-level parameter-mask formulation and is retained for conceptual context; the evaluated default is the Option-A gradient-routing implementation in Section 3.

to an edit-relevant subspace. However, the evaluated SAE-LOCKED LORA implementation uses Option-A gradient routing rather than directly masking model parameters. In this implementation, selected SAE coordinates constrain the edit-loss gradient, while preservation objectives remain unrestricted. The figure is retained to illustrate the conceptual motivation behind selective updates, not as a representation of the evaluated architecture or its empirical guarantees.

B SUPPLEMENTARY RESULTS AND QUALITATIVE DIAGNOSTICS

This appendix provides complementary results and diagnostics that extend the main-paper analysis without duplicating figures or tables already shown in the main text. We include the detailed optimization procedure, external KnowEdit context, the ReCoE reasoning-type breakdown, the required per-edit audit schema, the fixed-budget analysis, and edit-level collateral-drift visualizations.

B.1 DETAILED OPTIMIZATION PROCEDURE

Algorithm 2 expands the per-edit optimization procedure. Feature attribution and adaptive mask construction are performed once for each request, after which the selected set remains fixed throughout optimization. Only the LoRA parameters are updated; the backbone and SAE remain frozen. The edit-loss gradient is routed through the selected SAE coordinates, whereas preservation gradients bypass the mask.

Per-edit optimization procedure

Require: Frozen model f_θ; frozen SAE (E, D) at site (ℓ, site); edit prompts $\mathcal{E}$ with targets $\mathcal{Y}^\star$; holdout prompts $\mathcal{H}$; LoRA rank r; coverage threshold ρ; bounds $K_{\min}, K_{\max}$; T steps.
Compute normalized SAE feature scores $\{s_j\}_{j=1}^{k}$ for the edit request using gradient attribution.

Sort features such that $s_{\pi_1} \geq s_{\pi_2} \geq \cdots \geq s_{\pi_k}$.
Select

$$K_e = \min \left\{ K \in [K_{\min}, K_{\max}] : \frac{\sum_{q=1}^{K} s_{\pi_q}}{\sum_{j=1}^{k} s_j + \epsilon} \geq \rho \right\}.$$

Form the adaptive mask $m_j = \mathbb{I}[j \in \{\pi_1, \ldots, \pi_{K_e}\}]$ and selected set $S_e = \{j : m_j = 1\}$.
Initialize LoRA parameters $\phi = \{A, B\}$; keep θ and (E, D) **frozen**.
for $t = 1, \ldots, T$ **do**
Sample an edit minibatch from $(\mathcal{E}, \mathcal{Y}^\star)$ and a holdout minibatch from $\mathcal{H}$.

Zero adapter gradients and enable the masked activation-space backward hook.
Backpropagate $\mathcal{L}_{\text{edit}}$ through the selected SAE coordinates to obtain g_{edit}.
Bypass the edit mask and backpropagate the preservation objective to obtain g_{preserve}.
Set $\nabla_\phi \leftarrow g_{\text{edit}} + g_{\text{preserve}}$ and take one AdamW step.

end for
Ensure: Edited adapter ϕ and audit record $\mathcal{R}_e$ containing the selected set, attribution scores, coverage, stability, intervention response, and collateral-drift statistics.

Algorithm 2: **Detailed per-edit optimization for SAE-LOCKED LORA**

B.2 EXTERNAL KNOWEDIT CONTEXT

Table 18 reports external KnowEdit reference values for additional context. These numbers are *not* treated as matched comparisons with the reproduced results in the main paper because model variant, benchmark version, implementation, aggregation, and evaluator details may differ.

Table 18: External KnowEdit reference results on Llama2-7b-chat. Insert, Modify, and Erase correspond to reported edit-success values on WikiDatarecent, ZsRE, and Sanitation. These values provide context only and are not paired comparisons with the reproduced protocol.

External method	Insert ↑	Modify ↑	Erase ↑	Source/version
ROME (Meng et al., 2022)	85.08	96.57	**85.00**	KnowEdit (Llama2-7b-chat)
MEMIT (Meng et al., 2023)	85.32	83.07	48.75	KnowEdit (Llama2-7b-chat)
MEND (Mitchell et al., 2022a)	76.88	96.74	0.00	KnowEdit (Llama2-7b-chat)
SERAC (Mitchell et al., 2022b)	**98.68**	**99.67**	0.00	KnowEdit (Llama2-7b-chat)
FT-L	71.18	54.65	0.00	KnowEdit (Llama2-7b-chat)

B.3 RECOE REASONING-TYPE BREAKDOWN

The canonical matched ReCoE baseline frontier appears in Table 4. Appendix A.3 reports the complete six-category eligibility and observed-flip checksum, separating the instance-weighted micro `correct_flip` from the category macro without mixing checkpoints or prompting protocols.

B.4 QUALITATIVE PER-EDIT AUDIT RECORDS

Every attempted edit produces a canonical machine-readable audit JSON. The schema below identifies the fields retained for behavioral, mechanistic, and collateral-change analysis; displayed case studies can be rendered directly from the same record rather than hand-copied.

Table 19: **Required per-edit audit record.** Fields remain separate because they answer different questions; passing the record is not a safety or causal-completeness certificate.

Field group	Retained evidence
Request identity	Dataset, split, edit ID, prompt, base answer, target, model/tokenizer/SAE hashes, code revision, evaluator revision, and seed
Allocation	S_e, K_e, selected scores, realized coverage C_e, $K_{\min}/K_{\max}$ flags, and saturation status
Behavior	Direct success, paraphrase transfer, locality outcomes, and the exact decoding/evaluation settings
Mechanism checks	Paraphrase/seed mask overlap, selected-feature ablation response, and any automatically produced context summaries
Collateral change	KL drift plus Drift_{S_e} and $\mathrm{Drift}_{\bar{S}_e}$ on identified holdout prompts

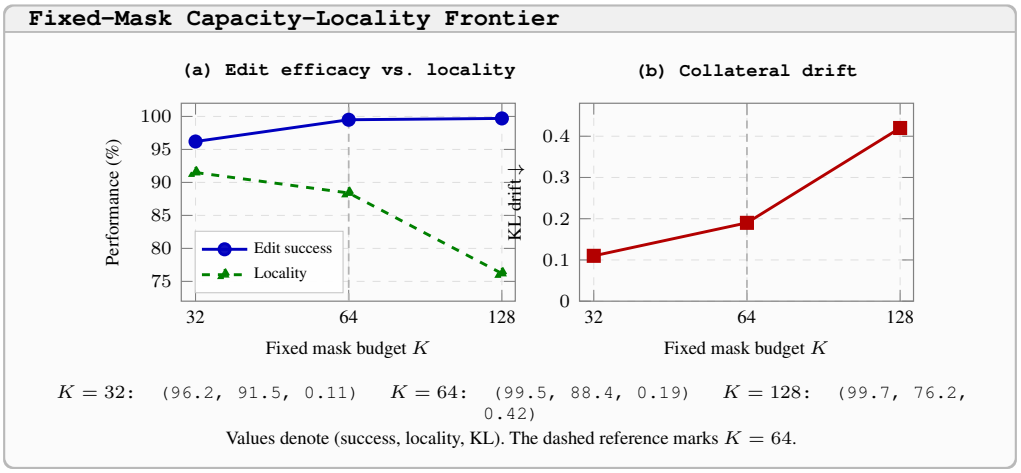

Figure 5: **Fixed-mask capacity–locality frontier. (a)** Edit success approaches saturation as the fixed mask grows, whereas locality deteriorates beyond $K = 64$. **(b)** KL drift increases with mask capacity. Increasing K from 64 to 128 yields only 0.2 additional success points, while locality decreases by 12.2 points and KL rises from 0.19 to 0.42. The main method avoids a single fixed operating point by selecting an edit-specific budget K_e.

B.5 Fixed-budget analysis

Figure 5 isolates the capacity–locality trade-off induced by a fixed mask budget. Increasing K rapidly saturates edit success, but progressively weakens locality and increases KL drift. This motivates request-adaptive allocation rather than assigning the same representational capacity to every edit.

B.6 Collateral-drift structure

Figure 6 resolves locality at the level of individual edit–probe pairs. Rows correspond to the same 100 sampled CounterFact edits in both panels, while columns denote held-out locality splits. Vanilla LoRA exhibits broad and heterogeneous collateral drift. In contrast, SAE-Locked LoRA shifts most edit–split pairs toward low KL, although a small high-drift tail remains. The shared scale makes this difference directly comparable across panels.

C Related Work

C.1 Locate-and-edit factual association methods

Knowledge editing (a.k.a. *model editing*) aims to modify a pretrained LM so that a targeted fact/behavior changes while the rest of the model remains stable (Yao et al., 2023; Wang et al., 2025b; Zhang et al., 2024). A major line of work localizes the internal computation responsible for a factual association and then applies a constrained parameter update. **ROME** uses causal tracing to identify a small set of mid-layer MLP weights mediating a subject–relation–object prediction and

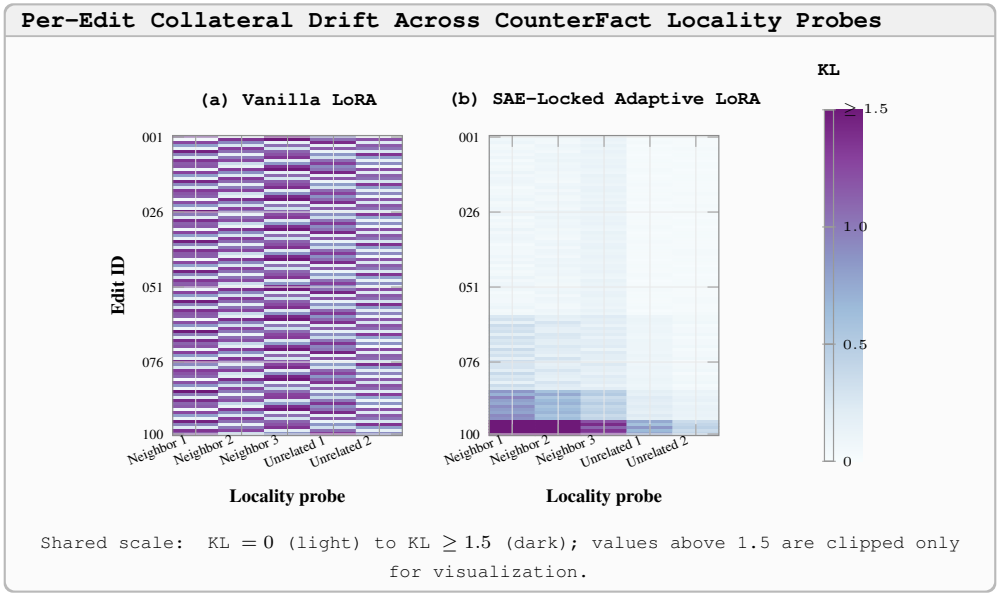

Figure 6: **Edit-level collateral-drift structure.** Each row represents the same sampled CounterFact edit in both panels, and each column corresponds to one held-out locality split. **(a)** Vanilla LoRA exhibits widespread and heterogeneous collateral drift across both neighboring and unrelated probes. **(b)** SAE-LOCKED LORA concentrates most edit–probe pairs near low KL, while retaining a small high-drift tail. Both panels use an identical color scale clipped at KL ≥ 1.5 for visualization; the underlying values are unchanged. The plot therefore visualizes the *distribution and concentration* of collateral drift rather than only its dataset-level mean.

performs a rank-1 style update at those locations; it is evaluated with the **CounterFact** benchmark that emphasizes edit success and locality (Meng et al., 2022). **MEMIT** extends this locate-and-edit paradigm to mass insertion of many facts by distributing explicit updates across multiple layers while aiming to preserve locality (Meng et al., 2023). Complementary approaches learn editors or external memories to produce local updates from an edit example, such as hypernetwork-based **KnowledgeEditor** (De Cao et al., 2021), gradient-decomposition editors (**MEND**) (Mitchell et al., 2022a), and semi-parametric memory-based editing (**SERAC**) (Mitchell et al., 2022b). More recent *lifelong-editing* analyses highlight persistent trade-offs among reliability, generalization, and locality, and propose mechanisms such as dynamic layer selection (**WilKE**) or alternative memory formulations (**WISE**) (Hu et al., 2024; Wang et al., 2024a). *How we differ:* rather than only localizing *where* to edit in parameter space, we constrain *which directions* a parameter-efficient update may take by locking LoRA updates to a sparse-feature mask, targeting improved locality under a fixed update budget.

C.2 BENCHMARKS FOR EDITING AND PROPAGATION

Evaluation has evolved from single-fact accuracy to **multi-dimensional protocols** that test generality, locality, and side effects (Wang et al., 2025b; Zhang et al., 2024). **CounterFact** remains a standard for assessing factual edits and unintended behavior changes (Meng et al., 2022), while **CounterFact+** specifically targets detection of edit failures and improved specificity stress tests (Hoelscher-Obermaier et al., 2023). Beyond single-hop factuality, **RippleEdits** formalizes *ripple effects*—the implied changes and invariances that should follow from an edit—and shows that prominent editors often fail to consistently integrate edits with related knowledge (Cohen et al., 2024). **MQuAKE** evaluates multi-hop question answering after edits to probe propagation through compositional reasoning chains (Zhong et al., 2023). **ReCoE** (Reasoning-based Counterfactual Editing) further operationalizes propagation via counterfactual reasoning tasks spanning multiple reasoning schemes and provides diagnostic analysis of why current editors struggle (Hua et al., 2024a). **KnowEdit** broadens coverage by benchmarking representative editing families under a

unified suite and analysis (Zhang et al., 2024), and **ConceptEdit** moves evaluation beyond instance edits to concept-level modifications (Wang et al., 2024b). Recent benchmark efforts also push toward broader, open-domain and ripple-aware settings (e.g., **UniEdit**) (Chen et al., 2025). *How we differ:* our primary contribution is not a new benchmark, but our method is designed to directly target the propagation/locality failure modes exposed by RippleEdits, MQuAKE, and ReCoE by restricting updates to a small, feature-addressable subspace.

C.3 SAE-BASED FEATURE LOCALIZATION AND EDITING

Sparse autoencoders (SAEs) have become a practical tool for decomposing LM activations into sparse, more monosemantic latent features, enabling automated interpretability and finer-grained causal analysis (Huben et al., 2024; Gao et al., 2025). Building on this, *sparse-feature-circuit* work uses SAE features as units for discovering causally implicated subnetworks and performing targeted interventions (Marks et al., 2025). Most closely related, **MicroEdit** leverages an SAE to disentangle knowledge representations and then performs neuron-level localization/scope control for lifelong editing (Wang et al., 2025a). *How we differ:* we use SAEs not primarily to switch or gate neurons, but to define a *feature mask* that constrains gradient flow for a LoRA update; i.e., the SAE features become the coordinate system that determines which update directions are permitted.

C.4 PEFT / LoRA

Parameter-efficient fine-tuning (PEFT) methods reduce adaptation cost by updating a small number of parameters (e.g., low-rank adapters) while freezing the backbone (Wang et al., 2025b). **LoRA** is a dominant PEFT mechanism that injects trainable low-rank matrices into linear layers (Hu et al., 2022), with widely used extensions such as **QLoRA** (quantized finetuning) (Dettmers et al., 2023) and **DoRA** (weight-decomposed low-rank adaptation) (Liu et al., 2024). However, PEFT alone does not guarantee *locality* or *minimal collateral change*; small parameter counts can still induce broad representational shifts, which is central in editing-specific evaluations (Hoelscher-Obermaier et al., 2023; Cohen et al., 2024). *How we differ:* we treat LoRA as the optimization vehicle but introduce an explicit SAE-feature constraint so that PEFT updates are not merely small, but also *subspace-restricted* in a feature-addressable manner.

C.5 GRADIENT PROJECTION IN CONTINUAL LEARNING

Gradient Projection Memory (GPM) derives low-rank bases from prior-task activations and projects new-task gradients away from those protected historical subspaces (Saha et al., 2021). PEGP transfers this principle to parameter-efficient continual learning, including LoRA adapters (Qiao et al., 2025). These methods establish gradient projection as a general interference-control tool. Our setting is different: we do not retain a subspace for past tasks, and we do not project the update away from it. For each edit request, we attribute the current edit loss in a frozen SAE coordinate system and route the edit-loss gradient through the smallest bounded set that covers the requested attribution mass. Preservation gradients are accumulated separately without the mask. The contribution is therefore the per-request sparse coordinate construction, its integration with editing-specific locality objectives, and the associated diagnostics rather than gradient projection in the abstract.

D LIMITATIONS AND BROADER IMPACT

Dependence on SAE quality and hook locations. Our approach assumes that the **SAE** provides a useful sparse feature basis at the chosen attachment site(s). In practice, **SAE quality** can vary with the activation distribution, normalization choices, dictionary size, sparsity strength, and the specific layer/site where activations are collected (Huben et al., 2024; Gao et al., 2025). If the SAE features are poorly learned (e.g., high reconstruction error or weak sparsity), then feature selection may become noisy and the resulting mask may not correspond to a stable causal substrate for the edit. Similarly, the choice of **hook locations** matters: an SAE trained on *mid-layer MLP outputs* may capture features relevant to factual association, while other edits (e.g., style, safety behavior, or multi-step reasoning) may rely on different sites. We partially address this by reporting sensitivity to layer/site choices and by including a multi-layer SAE variant in ablations, but the best hook selection remains model- and task-dependent.

Distributed knowledge and adaptive mask capacity. A core design choice is restricting edits to a **sparse, edit-specific feature subset**. Rather than using a fixed mask size for all edits, our adaptive mask selects the smallest number of SAE features whose cumulative attribution mass exceeds a threshold ρ, subject to bounds $K_{\min}$ and $K_{\max}$. This can still fail when the target behavior is distributed across many SAE features or when the edit must be integrated into broader computations that are not well represented by the selected subset. In such cases, the adaptive rule may still underfit: the model can succeed on direct rewrite prompts but fail to generalize or propagate the edit through reasoning chains (e.g., **ReCoE**) (Hua et al., 2024a; Zhong et al., 2023; Cohen et al., 2024). Increasing the attribution threshold or $K_{\max}$ can mitigate this, but it also relaxes the directional constraint and may increase collateral changes. Thus, our method exposes a *capacity–locality trade-off*, which we quantify through threshold sweeps, fixed-mask comparisons, and propagation/locality frontier analysis.

Theoretical limits of adaptive mask selection. **Adaptive mask selection** reduces the brittleness of a strict fixed-K mask, but it does not eliminate all theoretical limitations. Attribution scores may miss weak but jointly necessary features, overemphasize features that correlate with the edit prompt rather than causally support the desired behavior, or select redundant features when the SAE dictionary contains split or overlapping latents. Consequently, the selected feature set is not guaranteed to be causally complete, globally optimal, or sufficient for all edit types. The bounds $K_{\min}$ and $K_{\max}$ also impose an explicit *capacity–locality trade-off*: small bounds may underfit diffuse edits, while large bounds may admit collateral drift. We mitigate this limitation through adaptive-threshold sweeps, fixed-K baselines, and propagation/locality frontier analysis, but fully uncertainty-aware or causally certified mask selection remains future work.

Computational and memory constraints. Our implementation requires scoring and storing attribution statistics over a large SAE dictionary; in our main setting this corresponds to a **32,768-dimensional SAE feature space** for a **7B-parameter model**. Although only LoRA parameters are optimized during editing, SAE training, adaptive feature scoring, mask selection, and drift measurement require additional forward/backward passes and activation storage at the chosen hook site. All reported experiments were run on **A100-class GPUs** with sufficient memory for the 7B backbone, LoRA adapters, SAE forward passes, and minibatched holdout prompts. The one-time SAE training cost and the per-edit feature-scoring overhead are therefore practical but non-negligible, and scaling to larger models, larger dictionaries, or multi-hook SAEs may require *activation checkpointing*, *mixed precision*, *feature pre-filtering*, or *distributed computation*.

Security, dual-use risks, and safeguards. Constraining edits in an interpretable feature space can improve reliability and enable automatic auditing, but it may also lower the barrier to targeted modification of model behavior. In adversarial settings, **feature-level control** could be misused to insert specific misinformation, weaken refusal behavior, bias model outputs for selected inputs, or create backdoor-like behaviors that are difficult to detect using coarse aggregate metrics. Adaptive mask selection may further increase this risk by allocating more feature capacity to difficult or distributed edits. To reduce these risks, we recommend releasing only *scope-limited and evaluation-oriented code*, logging every edit request together with its selected feature mask, adaptive mask size K_e, attribution scores, and drift record, restricting arbitrary editing interfaces for high-capability models, and evaluating edited models on safety, refusal, and misuse-oriented regression suites before deployment. We also recommend caution in releasing full SAE dictionaries for highly capable models, since such dictionaries may make targeted interventions easier. Thus, feature-level audit records should be treated as **diagnostic aids**, not as sufficient guarantees of safety.

Intended positive use. The intended use of this work is to improve the reliability of model editing by reducing collateral behavior change, making edits more measurable, and providing **feature-level diagnostics** for debugging and auditing. Adaptive masks are intended to make the edit constraint less brittle by allowing concentrated edits to remain sparse while giving more capacity to diffuse edits when needed. These benefits are most appropriate in controlled research or deployment settings where edits are reviewed, logged, and evaluated before release. Responsible use therefore requires combining feature-masked editing with standard safety evaluations, broad regression testing, and monitoring after deployment.

E NOTATION AND DEFINITIONS

Table 20: Notation used in the main text and appendix. Fixed K is reserved for ablations; the main method uses the adaptive edit-specific mask size K_e.

Symbol	Meaning
f_θ	Pretrained language model with parameters θ.
θ	Frozen base model parameters.
ϕ	Trainable LoRA parameters applied to the frozen backbone.
θ'	Edited model parameters after applying LoRA, $\theta' = \theta \oplus \phi$.
$\Delta\theta$	Conceptual parameter change from editing, implemented through LoRA parameters ϕ.
x	Input prompt or token sequence.
y	Output token or sequence.
$p_\theta(y \mid x)$	Model distribution over outputs under parameters θ.
e	Edit request specifying a desired behavior change.
$\mathcal{E}$	Edit prompt set associated with edit request e.
$\mathcal{Y}^\star$	Target outputs for edit prompts $\mathcal{E}$.
$\mathcal{H}$	Holdout prompt set used for locality regularization and evaluation.
ℓ	Transformer layer index where the SAE is attached.
$h_\ell(x)$	Activation vector at the chosen hook site for input x, with $h_\ell(x) \in \mathbb{R}^d$.
$\tilde{h}$	Normalized activation, e.g., $\tilde{h} = (h - \mu)/(\sigma + \epsilon)$.
$E(\cdot)$	SAE encoder mapping normalized activations to sparse latents.
D	SAE decoder mapping sparse latents to reconstructed activations.
z	SAE latent vector, $z = E(\tilde{h}) \in \mathbb{R}^k$.
$\hat{h}$	SAE reconstruction, $\hat{h} = D(z)$.
k	Number of SAE features, i.e., dictionary size.
j	SAE feature index, $j \in \{1, \ldots, k\}$.
s_j	Attribution score for SAE feature j.
ρ	Attribution-coverage threshold used for adaptive mask selection.
$K_{\min}, K_{\max}$	Lower and upper bounds on the adaptive mask size.
K_e	Edit-specific adaptive mask size selected for edit e.
m	Binary adaptive feature mask, $m \in \{0, 1\}^k$.
S_e	Adaptively selected feature set for edit e, $S_e = \{j : m_j = 1\}$.
$\bar{S}_e$	Complement of the selected feature set, $\bar{S}_e = \{1, \ldots, k\} \setminus S_e$.
C_e	Realized attribution coverage of the selected mask for edit e.
K	Fixed mask size used only in fixed-K ablations.
P_m	Diagonal feature-mask projector, $P_m = \mathrm{diag}(m)$.
$\Delta z(x)$	SAE latent drift induced by editing, $\Delta z(x) = z(x; \phi) - z(x; 0)$.
$\Delta(\theta; x)$	Scalar edit signal, such as the target-vs-base logit gap.
$\ell(\theta; x, y^\star)$	Edit loss for target $y^\star$ on prompt x.
W	Frozen weight matrix of a linear layer adapted by LoRA.
ΔW	LoRA low-rank update applied to W, where $W' = W + \Delta W$.
r	LoRA rank.
α	LoRA scaling factor.
$\theta \oplus \phi$	Frozen backbone θ with LoRA parameters ϕ applied.
$\mathcal{L}_{\text{edit}}$	Edit loss optimized on $\mathcal{E}$.
$\mathcal{L}_{\text{KL}}$	KL behavior-preservation regularizer on $\mathcal{H}$.
$\mathcal{L}_{\text{drift}}$	Feature-space drift penalty on unselected features $\bar{S}_e$.
$\mathcal{L}_{\text{wd}}$	LoRA parameter norm regularizer.
β, γ, η	Weights for KL, drift, and weight-decay regularization.
T	Number of optimization steps per edit.

F DETAILED SAE TRAINING

This appendix details how we train **sparse autoencoders (SAEs)** on frozen model activations, including *layer/site selection*, *activation collection*, *normalization*, *architectural choices*, *sparsity tuning*, and *diagnostic measurements*. The SAE is trained once, frozen, and then used as a **sparse feature coordinate system** for attribution, adaptive mask selection, and drift measurement in **SAE-Locked Adaptive LoRA**. Our procedure follows established SAE practice in language models (Huben et al., 2024; Gao et al., 2025).

F.1 LAYER AND SITE CHOICE

We attach SAEs to internal activations $h_\ell(x)$ at one or more fixed transformer sites. In the main Llama-2-7b-chat-hf configuration, the default site is the output of transformer block 16 (1-based indexing), immediately after the MLP `down_proj` and before residual addition. This **MLP output (post-FFN)** site is motivated by prior evidence that mid-layer feed-forward activations mediate many factual associations and are commonly used as edit sites in locate-and-edit methods such as **ROME** (Meng et al., 2022). We use a *mid-layer* because early layers tend to encode more lexical or syntactic information, whereas later layers are more tightly coupled to output logits and may mix multiple behaviors, making sparse dictionary learning less stable. As an ablation, we also train SAEs at multiple layers, such as three evenly spaced layers, to test robustness to hook placement.

Selection rule. For a **single-layer SAE**, we choose ℓ from the middle third of the model, i.e.,

$$\ell \in \left[\left\lfloor \frac{L}{3} \right\rfloor, \left\lfloor \frac{2L}{3} \right\rfloor \right] \tag{8}$$

where L is the total number of transformer layers. We then run a lightweight development check over candidate layers by measuring reconstruction error, sparsity, and feature stability on held-out activations, and select the layer with the best *reconstruction–sparsity trade-off*.

F.2 ACTIVATION COLLECTION PIPELINE

We collect activation vectors by running the frozen base model f_θ over public text or benchmark corpora and recording $h_\ell(x)$ at the chosen site(s). To avoid storing full hidden-state tensors for long sequences, we use a **streaming pipeline**.

Inputs. For the main SAE, we sample 100 million token activations from the RedPajama-1T training split, tokenize with the base-model tokenizer, and stream fixed-length sequences through the frozen model. Benchmark test prompts are not used to train the SAE. The same frozen dictionary is used for CounterFact, KnowEdit, and ReCoE; cross-distribution diagnostics are reported in Table 15.

Recorded activations. For each sequence x, we record activations either at all token positions or at a subsampled set of token positions to reduce storage and increase diversity. Let $h_{\ell,t}(x) \in \mathbb{R}^d$ denote the activation at layer/site ℓ and token position t. The activation dataset is

$$\mathcal{A} = \{h_{\ell,t}(x)\}_{x,t} \tag{9}$$

Train/validation split. We create a held-out activation validation set $\mathcal{A}_{\text{val}}$ by reserving a fixed subset of sequences that is disjoint from the SAE training sequences. This validation set is used for hyperparameter tuning and for reporting reconstruction, sparsity, and usage diagnostics.

F.3 NORMALIZATION

We apply **per-dimension normalization** to stabilize SAE training and make sparsity coefficients comparable across layers. Let $\mathcal{A}_{\text{stat}} \subset \mathcal{A}$ denote the subset of activation vectors used to estimate normalization statistics. We compute the per-dimension running mean and standard deviation as

$$\mu = \mathbb{E}_{h \sim \mathcal{A}_{\text{stat}}}[h], \qquad \sigma = \sqrt{\mathbb{E}_{h \sim \mathcal{A}_{\text{stat}}}\left[(h-\mu)^{\odot 2}\right]} \tag{10}$$

where $\mu, \sigma \in \mathbb{R}^d$ and $(\cdot)^{\odot 2}$ denotes elementwise squaring. Each activation vector is normalized elementwise as

$$\tilde{h} = \frac{h - \mu}{\sigma + \epsilon} \tag{11}$$

We store (μ, σ) and reuse them at inference time when extracting SAE latents for feature scoring, adaptive mask selection, and drift measurement. In ablations, we compare this explicit standardization against no normalization and layer-normalized activations; explicit standardization typically improves reconstruction stability and feature sparsity, consistent with prior SAE scaling studies (Gao et al., 2025).

F.4 SAE ARCHITECTURE

We use a standard **single-hidden-layer sparse autoencoder**. The encoder maps normalized activations to sparse latents,

$$z = E(\tilde{h}) = \sigma_{\text{act}}\Big(W_E\tilde{h} + b_E\Big), \qquad z \in \mathbb{R}^k \tag{12}$$

and the decoder reconstructs the normalized activation:

$$\hat{h} = D(z) = W_D z + b_D, \qquad \hat{h} \in \mathbb{R}^d \tag{13}$$

Here, $W_E \in \mathbb{R}^{k\times d}$, $W_D \in \mathbb{R}^{d\times k}$, and $\sigma_{\text{act}}(\cdot)$ is a sparsity-compatible nonlinearity. Our default SAE uses **ReLU activations** with an ℓ_1 sparsity penalty, matching the training objective used in the main method.

Dictionary size. We set the number of SAE features k proportional to the activation dimension d:

$$k = \rho_{\text{dict}} d, \qquad \rho_{\text{dict}} \in \{4, 8, 16\} \tag{14}$$

Larger dictionaries can improve reconstruction and disentanglement but increase memory cost and expand the search space for feature attribution and adaptive mask selection (Gao et al., 2025). We therefore report sensitivity to ρ_{dict}.

Optimization. We optimize SAE parameters with `Adam` for 100,000 steps with batch size 4,096. The main dictionary has 32,768 ReLU features and uses $\lambda = 3 \times 10^{-3}$. After training, SAE parameters are **frozen** and are never updated during editing.

F.5 SPARSITY COEFFICIENT SWEEP

We tune the **sparsity coefficient** λ in the SAE objective:

$$\min_{E,D} \; \mathbb{E}_{h\sim\mathcal{A}}\left[\left\|\tilde{h} - D(E(\tilde{h}))\right\|_2^2\right] + \lambda\,\mathbb{E}_{h\sim\mathcal{A}}\left[\left\|E(\tilde{h})\right\|_1\right] \tag{15}$$

We sweep λ over a log grid, e.g.,

$$\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\} \tag{16}$$

and select the value that gives the best reconstruction–sparsity trade-off on $\mathcal{A}_{\text{val}}$. We report the full sweep curve in Figure 7, including reconstruction MSE, average active feature count, and stability across random seeds.

F.6 RECONSTRUCTION QUALITY DIAGNOSTICS

We monitor **reconstruction quality** using validation MSE,

$$\text{MSE} = \mathbb{E}_{h\sim\mathcal{A}_{\text{val}}}\left[\left\|\tilde{h} - \hat{h}\right\|_2^2\right] \tag{17}$$

and explained variance,

$$R^2 = 1 - \frac{\mathbb{E}_{h\sim\mathcal{A}_{\text{val}}}\left[\left\|\tilde{h} - \hat{h}\right\|_2^2\right]}{\mathbb{E}_{h\sim\mathcal{A}_{\text{val}}}\left[\left\|\tilde{h} - \mathbb{E}[\tilde{h}]\right\|_2^2\right]} \tag{18}$$

We also report cosine similarity between $\tilde{h}$ and $\hat{h}$ averaged over $\mathcal{A}_{\text{val}}$, along with the distribution of per-example reconstruction error. Figure 8 plots the reconstruction-error histogram for the selected SAE configuration.

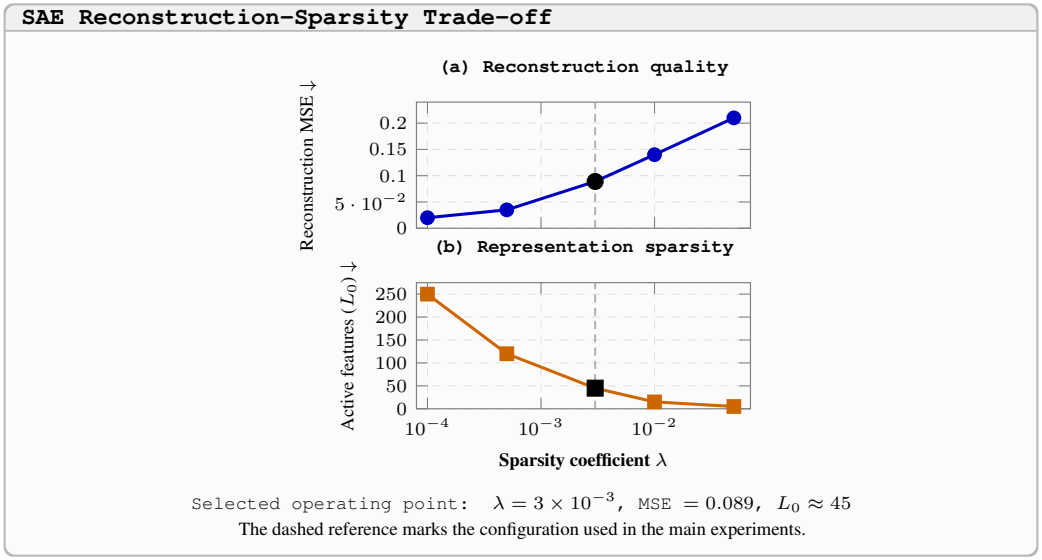

Figure 7: **SAE reconstruction–sparsity trade-off.** Increasing the sparsity coefficient λ reduces the number of active SAE features but increases reconstruction error. **(a)** Reconstruction MSE rises from 0.020 to 0.210 across the sweep. **(b)** Average activation sparsity improves from approximately 250 to 5 active features per token. We use $\lambda = 3 \times 10^{-3}$, yielding MSE $= 0.089$ and approximately 45 active features, as a sparse but sufficiently expressive operating point for edit-specific attribution and adaptive mask selection.

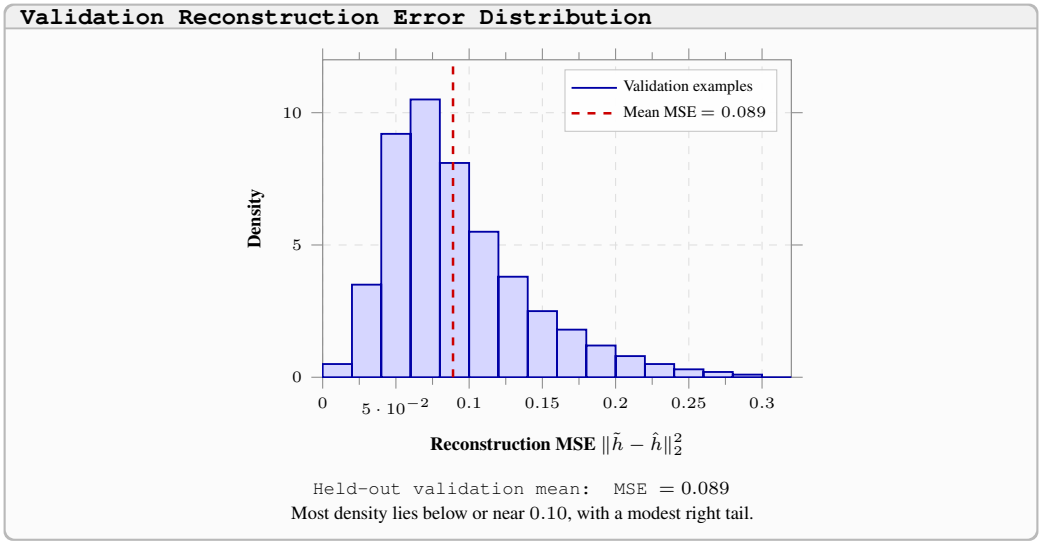

Figure 8: **Reconstruction error distribution.** Histogram of per-example reconstruction MSE on the held-out validation set $\mathcal{A}_{\text{val}}$. The distribution is unimodal with a modest right tail, and the mean MSE is 0.089, consistent with the selected SAE configuration. Most of the density is concentrated below or near 0.10, indicating stable reconstruction for most inputs.

F.7 Feature Sparsity and Usage

We quantify sparsity using **activation density**,

$$\text{Density}(z) = \mathbb{E}\left[\frac{\|z\|_0}{k}\right] \tag{19}$$

and **average active feature count**,

$$\mathrm{ActiveCount}(z) = \mathbb{E}\left[\|z\|_0\right] \tag{20}$$

Here, $\|z\|_0$ counts nonzero entries after the SAE activation function. We also measure **feature usage balance** across the dataset using the empirical activation frequency

$$\pi_j = \mathbb{E}_x\left[\mathbb{I}\{z_j(x) > 0\}\right] \tag{21}$$

Highly imbalanced usage, such as many dead features or extremely frequent features, can indicate a suboptimal choice of λ, dictionary size k, or activation normalization. We filter configurations that exhibit poor reconstruction, excessive density, or severe feature collapse.

F.8 Monosemanticity diagnostics

As an informative diagnostic, we estimate whether SAE features behave *monosemantically*, meaning they respond to a coherent pattern rather than many unrelated ones, following recent interpretability work (Bricken et al., 2023; Huben et al., 2024). Because monosemanticity is not uniquely defined, we report **automatic proxy diagnostics** rather than manual labels.

Top-context coherence. For each feature j, we collect the top-M activating contexts and compute textual similarity among the extracted windows using an embedding model. Let $\mathbf{u}_{j,a}$ and $\mathbf{u}_{j,b}$ denote embeddings of two top-activating contexts for feature j. We summarize coherence as the average pairwise cosine similarity:

$$\mathrm{Coh}(j) = \frac{2}{M(M-1)} \sum_{a<b} \frac{\mathbf{u}_{j,a}^\top \mathbf{u}_{j,b}}{\|\mathbf{u}_{j,a}\|_2 \|\mathbf{u}_{j,b}\|_2} \tag{22}$$

Higher coherence suggests that the feature responds to a narrower semantic pattern.

Context clustering entropy. We cluster the top-activating contexts for feature j into C clusters and compute the normalized entropy of cluster assignments:

$$H_j = -\frac{1}{\log C} \sum_{c=1}^{C} p_{j,c} \log p_{j,c} \tag{23}$$

where $p_{j,c}$ is the fraction of top contexts assigned to cluster c. Lower entropy indicates concentration in fewer clusters and therefore more coherent feature behavior.

Ablation specificity. We measure how selectively ablating feature j affects the model's output distribution on contexts where z_j is high versus contexts where z_j is low. Let $\mathcal{X}_j^{\mathrm{high}}$ and $\mathcal{X}_j^{\mathrm{low}}$ denote high-activation and low-activation context sets for feature j. We define the specificity score

$$\mathrm{Spec}(j) = \mathbb{E}_{x \in \mathcal{X}_j^{\mathrm{high}}}\left[\mathrm{KL}\left(p_\theta(\cdot \mid x) \| p_\theta^{(-j)}(\cdot \mid x)\right)\right] - \mathbb{E}_{x \in \mathcal{X}_j^{\mathrm{low}}}\left[\mathrm{KL}\left(p_\theta(\cdot \mid x) \| p_\theta^{(-j)}(\cdot \mid x)\right)\right] \tag{24}$$

where $p_\theta^{(-j)}$ denotes the model distribution after clamping feature j at the SAE hook site. A larger value indicates that ablating the feature has a stronger effect when the feature is naturally active, supporting a narrower causal role.

These diagnostics are not definitive proofs of monosemanticity, but they provide transparent and reproducible evidence about feature coherence. They also support our use of SAE features as editing coordinates: if selected features are coherent, stable, and selectively causal, then adaptive feature masks provide a more interpretable and auditable constraint than raw neuron or parameter-level selection.

G Attribution and Adaptive Mask Selection Details

This appendix gives the full attribution and mask-selection details for **SAE-Locked Adaptive LoRA**. The main method uses *adaptive attribution-mass masking*: for each edit request, we score SAE features, sort them by relevance, and select the smallest feature set whose cumulative attribution mass exceeds a threshold ρ, subject to lower and upper bounds $K_{\min}$ and $K_{\max}$. Fixed-K masks are used only as ablations to study the *success–locality frontier*.

G.1 PRELIMINARIES

For an **edit request** e, let $\mathcal{E} = \{x_i\}_{i=1}^{n_E}$ denote the edit prompts and let $\mathcal{Y}^\star = \{y_i^\star\}_{i=1}^{n_E}$ denote the desired target tokens or target sequences. Let $h_\ell(x)$ be the activation at the selected hook site, such as the *MLP output* of layer ℓ, and let $\tilde{h}_\ell(x)$ be its normalized version. The frozen SAE encoder maps this activation to sparse latent features:

$$z(x) = E(\tilde{h}_\ell(x)) \in \mathbb{R}^k \tag{25}$$

Here, k is the number of SAE features and $j \in \{1, \dots, k\}$ indexes a feature. For token-level factual edits, $y^\star$ is often a single token; when the target is a sequence, we compute logits and losses by summing token-level terms over the target span. All feature scores are computed before LoRA optimization unless explicitly stated as *post-hoc diagnostics*.

G.2 FEATURE SCORING RULES

We consider three automatic scoring families. Each produces a scalar relevance score s_j for every SAE feature j. Scores are converted to nonnegative selection weights before mask construction, either by taking absolute values or by using magnitude-based definitions. In the main method, we use **gradient attribution** because it is edit-specific, cheaper than causal ablation, and empirically gives the strongest success/generalization trade-off.

G.2.1 LOGIT-DIFFERENCE ATTRIBUTION

Let $\mathcal{V}$ be the vocabulary, and let $\mathrm{logit}_\theta(v \mid x)$ denote the pre-softmax logit for token v at the prediction position for prompt x under the frozen base model. Let $y^{\mathrm{base}}(x)$ be the reference token representing the base behavior, such as the base model's current argmax token or a known undesired token. We define the **target-vs-base logit gap** as

$$\Delta_\theta(x) = \mathrm{logit}_\theta(y^\star \mid x) - \mathrm{logit}_\theta(y^{\mathrm{base}}(x) \mid x) \tag{26}$$

A simple alignment score is

$$s_j^{\mathrm{ld}} = |\mathbb{E}_{x \in \mathcal{E}} [z_j(x) \Delta_\theta(x)]| \tag{27}$$

When activation scale or prompt-specific offsets are a concern, we instead use a *centered correlation-style score*:

$$s_j^{\mathrm{ld}} = \left|\mathbb{E}_{x \in \mathcal{E}} \left[(z_j(x) - \bar{z}_j)(\Delta_\theta(x) - \overline{\Delta})\right]\right| \tag{28}$$

Here, $\bar{z}_j$ and $\overline{\Delta}$ are means over the edit prompts. **Logit-difference attribution** is computationally cheap because it only requires forward activations and logits, but it can select features correlated with the prompt rather than features causally involved in the desired edit.

G.2.2 GRADIENT ATTRIBUTION

Let $\ell(\theta; x, y^\star)$ denote the edit loss for prompt-target pair $(x, y^\star)$ under the frozen base model, typically the negative log-likelihood of the target token or target span. We compute the magnitude of the edit-loss gradient with respect to each SAE latent coordinate:

$$g_j(x) = \left| \frac{\partial \ell(\theta; x, y^\star)}{\partial z_j(x)} \right| \tag{29}$$

The default feature score is the **mean gradient magnitude** over edit prompts:

$$s_j^{\mathrm{grad}} = \mathbb{E}_{(x, y^\star) \in (\mathcal{E}, \mathcal{Y}^\star)} [g_j(x)] \tag{30}$$

As a robust alternative, we also consider *median aggregation*:

$$s_j^{\mathrm{grad}} = \mathrm{median}_{(x, y^\star)} [g_j(x)] \tag{31}$$

To reduce scale sensitivity across SAE features, we optionally normalize by the empirical standard deviation of the feature on a background set $\mathcal{B}$:

$$\tilde{s}_j^{\mathrm{grad}} = \frac{s_j^{\mathrm{grad}}}{\mathrm{Std}_{x \sim \mathcal{B}}[z_j(x)] + \epsilon} \tag{32}$$

Unless otherwise stated, the main experiments use mean gradient attribution with standard-deviation normalization. The normalized scores are then clipped below at zero and passed to adaptive mask selection.

G.2.3 CAUSAL ABLATION ATTRIBUTION

Causal ablation measures how much the edit signal changes when a feature is removed from the SAE representation. Let e_j denote the one-hot vector for feature j. We clamp feature j by setting

$$z^{(-j)}(x) = z(x) \odot (\mathbf{1} - e_j) \tag{33}$$

We decode the ablated latent as $\hat{h}^{(-j)} = D(z^{(-j)}(x))$ and substitute this reconstruction at the SAE hook site during a forward pass. Let $\Delta_\theta^{(-j)}(x)$ denote the resulting target-vs-base logit gap. The ablation score is

$$s_j^{\text{abl}} = \left| \mathbb{E}_{x \in \mathcal{E}} \left[\Delta_\theta(x) - \Delta_\theta^{(-j)}(x) \right] \right| \tag{34}$$

This score is more directly intervention-based than logit-difference or gradient attribution, but it is substantially more expensive because it requires additional forward passes for candidate features. We therefore use causal ablation primarily as a *post-hoc diagnostic* or on a pre-filtered candidate set, rather than as the default scoring rule.

G.3 ADAPTIVE MASK SELECTION

The main method uses **adaptive mask selection**. Given nonnegative feature scores $\{s_j\}_{j=1}^k$, we first sort features by descending score. Let π be a permutation such that

$$s_{\pi_1} \geq s_{\pi_2} \geq \cdots \geq s_{\pi_k} \tag{35}$$

Let $\rho \in (0, 1)$ be the attribution-coverage threshold. The adaptive budget for edit e is

$$K_e = \min \left\{ K \in [K_{\min}, K_{\max}] : \frac{\sum_{r=1}^{K} s_{\pi_r}}{\sum_{j=1}^{k} s_j + \epsilon} \geq \rho \right\} \tag{36}$$

The selected feature set and binary mask are then

$$S_e = \{\pi_1, \ldots, \pi_{K_e}\}, \qquad m_j = \mathbb{I}[j \in S_e] \tag{37}$$

This rule selects the smallest mask that covers the required attribution mass. Concentrated edits therefore use small masks, while diffuse edits can allocate more feature directions up to $K_{\max}$. The bounds $K_{\min}$ and $K_{\max}$ prevent degenerate masks: $K_{\min}$ ensures enough capacity for optimization, while $K_{\max}$ limits collateral drift from overly broad masks.

Degenerate cases. If all scores are zero or numerically negligible, we fall back to selecting the top $K_{\min}$ features after deterministic tie-breaking. If the attribution coverage threshold cannot be met before $K_{\max}$, we set $K_e = K_{\max}$. We record these cases because frequent saturation at $K_{\max}$ indicates that the edit may be too distributed for the chosen SAE hook site, dictionary size, or attribution rule.

G.4 FIXED-K MASKS AS AN ABLATION

Fixed-K selection is not the main method, but we include it as an ablation to study the role of mask capacity. Given scores $\{s_j\}_{j=1}^k$ and a fixed budget K, the fixed mask is

$$S_K = \{\pi_1, \ldots, \pi_K\}, \qquad m_j^{(K)} = \mathbb{I}[j \in S_K] \tag{38}$$

This ablation isolates the effect of mask size. Small fixed masks tend to preserve locality but may underfit diffuse edits. Large fixed masks improve edit success and sometimes propagation, but they can weaken the directional constraint and increase collateral drift. Adaptive masking avoids selecting one global value of K for all edits by choosing K_e separately for each edit according to attribution mass.

G.5 REDUNDANCY-AWARE ADAPTIVE SELECTION

The attribution-mass rule assumes that feature scores are additive and that high-scoring features provide nonredundant evidence. This may fail when the SAE dictionary contains split or overlapping

Adaptive mask-selection procedure

Require: Scores $\{s_j\}_{j=1}^k$, coverage threshold ρ, bounds $K_{\min}, K_{\max}$, numerical constant ϵ.
Convert scores to nonnegative weights, e.g., $s_j \leftarrow |s_j|$ when needed.
Sort indices such that $s_{\pi_1} \geq s_{\pi_2} \geq \cdots \geq s_{\pi_k}$ using deterministic tie-breaking.
Compute total attribution mass $T \leftarrow \sum_{j=1}^k s_j + \epsilon$.
if $T \leq \epsilon$ **then**
$K_e \leftarrow K_{\min}$.
else
$K_e \leftarrow K_{\max}$.
for $K = K_{\min}$ to $K_{\max}$ **do**
if $\sum_{r=1}^K s_{\pi_r}/T \geq \rho$ **then**
$K_e \leftarrow K$.
break.
end if
end for
end if
$S_e \leftarrow \{\pi_1, \ldots, \pi_{K_e}\}$.
$m_j \leftarrow \mathbb{I}[j \in S_e]$ for all j.
Ensure: Adaptive feature set S_e, binary mask m, and selected budget K_e.

Algorithm 3: **Adaptive attribution-mass mask selection**

Fixed-budget mask-selection procedure

Require: Scores $\{s_j\}_{j=1}^k$ and fixed feature budget K.
Sort indices j by descending s_j using deterministic tie-breaking.
Select the top-K feature set $S_K \leftarrow \{\pi_1, \ldots, \pi_K\}$.

Construct the binary mask $m_j^{(K)} \leftarrow \mathbb{I}[j \in S_K]$ for all j.
Ensure: Fixed-budget feature set S_K and binary mask $m^{(K)}$.

Algorithm 4: **Fixed-K mask selection for ablation**

features. As an optional variant, we use **redundancy-aware adaptive selection**. Let $F(S)$ be a set function combining attribution mass and redundancy penalty:

$$F(S) = \sum_{j \in S} s_j - \lambda_{\text{red}} \sum_{\substack{j,j' \in S \\ j<j'}} \text{sim}(j, j') \tag{39}$$

Here, $\text{sim}(j, j')$ can be the cosine similarity between decoder columns $D_{\cdot j}$ and $D_{\cdot j'}$, and $\lambda_{\text{red}} \geq 0$ controls the penalty. We greedily add features by marginal gain:

$$j^\star = \arg\max_{j \notin S} \left[s_j - \lambda_{\text{red}} \sum_{j' \in S} \text{sim}(j, j') \right] \tag{40}$$

We stop when the selected set reaches the attribution coverage threshold or hits $K_{\max}$. This variant can reduce redundant masks, but it introduces an additional hyperparameter and requires computing feature similarities. We therefore treat it as an optional analysis rather than the default.

G.6 Mask diagnostics

For each edit, we log the selected mask and several diagnostics:

$$K_e = |S_e|, \qquad C_e = \frac{\sum_{j \in S_e} s_j}{\sum_{j=1}^k s_j + \epsilon}, \qquad B_e = \mathbb{I}[K_e = K_{\min}] + \mathbb{I}[K_e = K_{\max}] \tag{41}$$

Redundancy-aware mask-selection procedure

Require: Scores $\{s_j\}$, similarity function $\mathrm{sim}(\cdot,\cdot)$, redundancy weight λ_{red}, coverage threshold ρ, bounds $K_{\min}, K_{\max}$, and numerical constant ϵ.
Initialize $S \leftarrow \emptyset$ and $T \leftarrow \sum_j s_j + \epsilon$.
while $|S| < K_{\max}$ **do**

Select

$$j^\star \leftarrow \arg\max_{j \notin S} \left[s_j - \lambda_{\mathrm{red}} \sum_{j' \in S} \mathrm{sim}(j, j') \right].$$

Update $S \leftarrow S \cup \{j^\star\}$.

if $|S| \geq K_{\min}$ **and** $\sum_{j \in S} s_j / T \geq \rho$ **then**
break.
end if
end while
Set $S_e \leftarrow S$.

Construct $m_j \leftarrow \mathbb{I}[j \in S_e]$ for all j.
Ensure: Redundancy-aware adaptive feature set S_e and binary mask m.

Algorithm 5: **Redundancy-aware adaptive mask selection**

Here, K_e is the selected adaptive budget, C_e is the realized attribution coverage, and B_e records whether the mask hits either bound. We report the mean and distribution of K_e across edits, the fraction of edits saturating at $K_{\max}$, and the relationship between K_e and edit difficulty. In practice, large K_e often indicates that the edit signal is distributed across many SAE features, while small K_e indicates a concentrated mechanism. These diagnostics are important because adaptive masking can improve locality only if the selected mask remains meaningfully sparse.

We also track **in-mask** and **out-of-mask latent drift** on holdout prompts. For $\Delta z(x) = z(x;\phi) - z(x;0)$, we compute

$$\mathrm{Drift}_{S_e} = \mathbb{E}_{x \sim \mathcal{H}} \left[\|\Delta z_{S_e}(x)\|_1 \right], \qquad \mathrm{Drift}_{\bar{S}_e} = \mathbb{E}_{x \sim \mathcal{H}} \left[\|\Delta z_{\bar{S}_e}(x)\|_1 \right] \tag{42}$$

A successful constrained edit should concentrate useful movement in S_e while suppressing drift in $\bar{S}_e$. We use these diagnostics to interpret locality improvements and to detect failure cases where the edit leaks outside the selected feature subspace.

G.7 Stability analysis

We evaluate **mask stability** under two sources of variation: prompt paraphrases and random seeds. Because adaptive masks can have different sizes across variants, we report both set-overlap and coverage-style metrics. This avoids overinterpreting Jaccard similarity when two masks differ in budget but still share the most important features.

G.7.1 Mask overlap across paraphrases

Let $\{\mathcal{E}^{(p)}\}_{p=1}^{P}$ be paraphrase groups for a given edit, such as CounterFact paraphrases. For each paraphrase group, we compute scores and construct an adaptive mask $S_e^{(p)}$ using the same threshold ρ and bounds $(K_{\min}, K_{\max})$. We quantify pairwise overlap using **Jaccard similarity**:

$$\mathrm{Jac}\big(S_e^{(p)}, S_e^{(q)}\big) = \frac{|S_e^{(p)} \cap S_e^{(q)}|}{|S_e^{(p)} \cup S_e^{(q)}|} \tag{43}$$

Because adaptive masks may differ in size, we also report **asymmetric retention** relative to a reference mask, usually the canonical rewrite prompt:

$$\mathrm{Ret}(S_e^{(p)}; S_e^{(0)}) = \frac{|S_e^{(p)} \cap S_e^{(0)}|}{|S_e^{(0)}|} \tag{44}$$

High Jaccard and high retention indicate that the selected features are tied to the edit semantics rather than surface wording. Low Jaccard but high retention can occur when a paraphrase includes the core reference features plus additional features, which is less concerning than low overlap in both metrics.

G.7.2 Mask overlap across seeds

For each edit, we repeat the full attribution and adaptive-selection pipeline with R random seeds, producing masks $\{S_e^{(r)}\}_{r=1}^{R}$. Seeds affect SAE initialization if the SAE is retrained, minibatch ordering, and any stochasticity in LoRA optimization. Pairwise seed stability is

$$\mathrm{Stab}_{\mathrm{seed}} = \frac{2}{R(R-1)} \sum_{r<r'} \mathrm{Jac}(S_e^{(r)}, S_e^{(r')}) \tag{45}$$

We additionally report the mean and standard deviation of selected budgets $\{K_e^{(r)}\}_{r=1}^{R}$ and the Spearman rank correlation between feature-score vectors across seeds. Stable masks should show both high set overlap and low variance in K_e.

G.7.3 Stability–performance relationship

Finally, we test whether mask stability predicts edit reliability. For each edit, we compute stability metrics, edit success, paraphrase generalization, locality, KL drift, out-of-mask drift, and ReCoE propagation when available. We report Pearson and Spearman correlations between stability and performance. The expected pattern is that unstable masks correlate with higher collateral drift and weaker propagation, suggesting that instability reflects either an underlocalized edit mechanism or ambiguity in the selected SAE feature basis. We treat this as a diagnostic rather than a guarantee: high mask stability does not prove causal completeness, and low stability does not necessarily imply failure if the edit can be represented by multiple redundant feature sets.

Reporting. For the main benchmarks, we report the distribution of selected budgets K_e, the fraction of edits reaching $K_{\max}$, paraphrase-level Jaccard overlap, seed-level Jaccard overlap, rank stability of feature scores, and stability-versus-locality scatter plots. These diagnostics make the adaptive mask selection auditable and distinguish successful sparse edits from cases where the method uses broad masks or unstable feature sets.

H SAE-Locked Adaptive LoRA Optimization

This appendix derives the optimization procedure used by **SAE-Locked Adaptive LoRA**. The main method is **Option A**: *masked feature-gradient routing*, where the edit-loss gradient is projected through an adaptively selected SAE feature mask. We also describe **Option B**: a harder feature-parameterized constraint that restricts LoRA updates to the span of selected SAE decoder directions. In both cases, the selected feature set is edit-specific: for each edit request e, we compute attribution scores, select an adaptive mask (S_e, m, K_e) using attribution-mass coverage, and then optimize only the LoRA parameters while keeping the backbone and SAE frozen. We also describe how the required Jacobian-vector products are computed efficiently by automatic differentiation without materializing full Jacobians.

H.1 Option A: masked feature-gradient routing

Fix a frozen SAE (E, D) trained at a chosen activation site. For a prompt x, let

$$h = h_\ell(x; \phi) \in \mathbb{R}^d \tag{46}$$

denote the raw activation at that site under LoRA parameters ϕ, with the backbone parameters fixed. Let $\mu, \sigma \in \mathbb{R}^d$ be the frozen per-coordinate normalization statistics used to train the SAE, and define

$$s = \sigma + \epsilon \mathbf{1}, \qquad S = \mathrm{diag}(s), \qquad \tilde{h} = S^{-1}(h - \mu) \tag{47}$$

The SAE therefore operates in normalized activation space:

$$z = E(\tilde{h}) \in \mathbb{R}^k, \qquad D(z) \approx \tilde{h} \tag{48}$$

Because $D(z)$ reconstructs $\tilde{h}$, its output must be mapped back to raw activation coordinates before being combined with h. Define the raw-space reconstruction operator

$$\mathcal{R}(v) = \mu + SD(v) = \mu + s \odot D(v) \tag{49}$$

For the current edit request e, adaptive mask selection returns a feature set S_e, a binary mask $m \in \{0,1\}^k$, and a selected budget $K_e = |S_e|$. The selected mask is treated as fixed throughout the optimization for that edit. We define the corresponding diagonal projector as

$$P_m = \mathrm{diag}(m) \in \mathbb{R}^{k \times k} \tag{50}$$

Option A preserves the model's raw forward activation while restricting the edit-loss gradient transmitted through the SAE site to the selected coordinates S_e.

Identity-preserving raw-space hook. A naive replacement of h by $D(E(\tilde{h}))$ would both mix normalized and raw coordinates and alter the forward computation because the SAE reconstruction is imperfect. Instead, let

$$\hat{h}_{\mathrm{raw}} = \mathcal{R}(z) = \mu + SD(z) \tag{51}$$

and let $\mathrm{sg}(\cdot)$ denote the stop-gradient operator. The activation passed to the remainder of the model is

$$h^{\mathrm{fwd}} = \hat{h}_{\mathrm{raw}} + \mathrm{sg}\Big(h - \hat{h}_{\mathrm{raw}}\Big) \tag{52}$$

In the forward pass, the stop-gradient term has value $h - \hat{h}_{\mathrm{raw}}$, and hence $h^{\mathrm{fwd}} = h$ exactly. The hook therefore leaves the model's forward predictions unchanged. During backpropagation, however, the residual term contributes no derivative, so gradients flow through the normalized SAE coordinates and their explicit raw-space de-normalization.

Adaptive masking in latent space. To restrict the edit-loss gradient to the selected feature set S_e, we define

$$z^{\mathrm{fwd}} = P_m z + \mathrm{sg}((I - P_m)z) \tag{53}$$

Numerically, $z^{\mathrm{fwd}} = z$, but its Jacobian with respect to z is

$$\frac{\partial z^{\mathrm{fwd}}}{\partial z} = P_m \tag{54}$$

The decoder output is then returned to raw activation space:

$$\hat{h}_{\mathrm{raw},m} = \mathcal{R}(z^{\mathrm{fwd}}) = \mu + SD(z^{\mathrm{fwd}}) \tag{55}$$

and the masked identity-preserving hook is

$$h^{\mathrm{fwd}} = \hat{h}_{\mathrm{raw},m} + \mathrm{sg}\Big(h - \hat{h}_{\mathrm{raw},m}\Big) \tag{56}$$

Equations 53– 56 jointly provide exact forward equivalence in raw activation space and masked backward transmission in SAE latent space.

To make the normalization geometry explicit, define

$$J_E(\tilde{h}) = \frac{\partial E(\tilde{h})}{\partial \tilde{h}}, \qquad J_D(z^{\mathrm{fwd}}) = \frac{\partial D(z^{\mathrm{fwd}})}{\partial z^{\mathrm{fwd}}} \tag{57}$$

The backward Jacobian induced by the hook is

$$\frac{\partial h^{\mathrm{fwd}}}{\partial h} = S\, J_D(z^{\mathrm{fwd}}) P_m J_E(\tilde{h})\, S^{-1} \tag{58}$$

The left factor S converts decoder perturbations from normalized coordinates back to raw activation units, while S^{-1} converts raw activation perturbations into the normalized coordinates consumed by the encoder. These factors cannot generally be omitted or cancelled, because the encoder and decoder Jacobians and the mask projector need not commute with the coordinate-wise scaling.

Projected gradient on LoRA parameters. Let

$$J_{h,\phi} = \frac{\partial h}{\partial \phi}, \qquad g_h = \frac{\partial \mathcal{L}_{\text{edit}}}{\partial h^{\text{fwd}}} \tag{59}$$

Using Equation 58, the edit-loss gradient transmitted through the hooked site is

$$\nabla_\phi \mathcal{L}_{\text{edit}} = J_{h,\phi}^\top S^{-1} J_E(\tilde{h})^\top P_m J_D(z^{\text{fwd}})^\top S\, g_h \tag{60}$$

Equivalently, defining

$$J_{z,\phi} = J_E(\tilde{h}) S^{-1} J_{h,\phi}, \qquad g_{z^{\text{fwd}}} = J_D(z^{\text{fwd}})^\top S g_h \tag{61}$$

gives the compact form

$$\nabla_\phi \mathcal{L}_{\text{edit}} = J_{z,\phi}^\top P_m g_{z^{\text{fwd}}} \tag{62}$$

Thus, Option A inserts the projector P_m into the latent-space vector–Jacobian product while retaining the correct conversion between normalized SAE coordinates and raw model activations. In the evaluated configuration, the trainable layer-16 `down_proj` produces the hooked activation, so every trainable edit-loss path crosses this projection. For adapter placements with bypass paths, the expression governs only the gradient contribution passing through the hooked site.

Total objective with unmasked regularizers. The full optimization objective is

$$\mathcal{L}_{\text{total}}(\phi) = \mathcal{L}_{\text{edit}}(\phi) + \beta \mathcal{L}_{\text{KL}}(\phi) + \gamma \mathcal{L}_{\text{drift}}(\phi) + \eta \mathcal{L}_{\text{wd}}(\phi) \tag{63}$$

Its adapter gradient is accumulated as

$$\nabla_\phi \mathcal{L}_{\text{total}} = \underbrace{\nabla_\phi \mathcal{L}_{\text{edit}}}_{\text{masked through } P_m} + \beta \nabla_\phi \mathcal{L}_{\text{KL}} + \gamma \nabla_\phi \mathcal{L}_{\text{drift}} + \eta \nabla_\phi \mathcal{L}_{\text{wd}} \tag{64}$$

Only the edit-loss gradient is projected through P_m. The KL and drift terms remain unmasked so that they can oppose collateral changes outside the edit-specific subspace.

Operationally, each optimizer step uses two backward passes. First, the mask is enabled and only $\mathcal{L}_{\text{edit}}$ is backpropagated. Second, the coordinate mask is disabled and the preservation objective

$$\mathcal{L}_{\text{preserve}} = \beta \mathcal{L}_{\text{KL}} + \gamma \mathcal{L}_{\text{drift}} + \eta \mathcal{L}_{\text{wd}} \tag{65}$$

is backpropagated without the P_m restriction. The resulting adapter gradients are summed before one AdamW update. Both passes use the same normalization and raw-space de-normalization defined in Equations 47 and 49. Applying a permanently masked hook to the summed loss would incorrectly project the preservation gradients and is not the evaluated implementation.

H.2 Efficient implementation: vector-Jacobian products without explicit Jacobians

Option A does not require explicitly forming the Jacobian $\partial z / \partial \phi$. Equation 60 is a **vector-Jacobian product**:

$$\left(\frac{\partial z}{\partial \phi}\right)^\top v, \qquad v = P_m \frac{\partial \mathcal{L}_{\text{edit}}}{\partial z^{\text{fwd}}} \tag{66}$$

Modern automatic differentiation computes this quantity with standard backpropagation. Thus, the computational cost is comparable to a normal backward pass through the model and SAE hook; full Jacobian matrices are never materialized.

Practical hook implementation and scope. During the edit backward pass, we compute $z = E(\tilde{h})$, form $z^{\text{fwd}} = P_m z + \text{sg}((I - P_m)z)$, decode and de-normalize $\hat{h}_{\text{raw},m} = \mu + (\sigma + \epsilon) \odot D(z^{\text{fwd}})$, and forward $h^{\text{fwd}} = \hat{h}_{\text{raw},m} + \text{sg}(h - \hat{h}_{\text{raw},m})$. The SAE parameters are frozen and excluded from the optimizer. In PyTorch, the stop-gradient operator is implemented with `.detach()`. The preservation pass bypasses the mask. In the main configuration, LoRA is attached only to the layer-16 MLP `down_proj` that produces the hooked activation; later layers are frozen, so no trainable edit-loss path bypasses the mask. Option A should not be interpreted as constraining arbitrary adapters placed downstream of or around the hook.

Multiple SAE sites. If SAEs are attached at multiple sites $\{\ell_s\}_{s=1}^{S}$, we compute an adaptive mask for each site or use a shared edit-level attribution budget distributed across sites. With independent site masks $m^{(s)}$, the effective projector is block-diagonal in the concatenated latent space:

$$P_{\text{multi}} = \text{diag}(P_{m^{(1)}}, P_{m^{(2)}}, \ldots, P_{m^{(S)}}) \tag{67}$$

The resulting adapter gradient is the sum of vector-Jacobian products from all hooked sites:

$$\nabla_\phi \mathcal{L}_{\text{edit}} \leftarrow \sum_{s=1}^{S} \left(\frac{\partial z^{(s)}}{\partial \phi} \right)^\top P_{m^{(s)}} \frac{\partial \mathcal{L}_{\text{edit}}}{\partial z^{(s),\text{fwd}}} \tag{68}$$

In the main experiments, we use the same hook configuration for all methods within a comparison, so any improvement is attributable to the adaptive feature constraint rather than to changing the attachment site.

Numerical considerations. We use *mixed precision* for the backbone and SAE forward/backward passes when appropriate, but keep masks in full precision to avoid numerical issues when gradients are small. The SAE forward is lightweight relative to the transformer, so it rarely dominates runtime. When memory is constrained, we use *activation checkpointing* for the backbone. We also clip LoRA gradients to stabilize optimization, especially for edits whose adaptive mask saturates at $K_{\max}$.

H.3 Option B: hard feature-parameterized constraint

Option B enforces a harder subspace constraint by restricting the LoRA-induced activation change to the span of the adaptively selected SAE decoder directions. This differs from **Option A**: Option A constrains the gradient pathway, while Option B constrains the parameterization of the update itself.

Consider an adapted linear map that produces the activation at the SAE hook site:

$$h = Wu, \qquad W \in \mathbb{R}^{d \times d_{\text{in}}}, \quad u \in \mathbb{R}^{d_{\text{in}}} \tag{69}$$

A standard LoRA update has the form

$$\Delta W = \frac{\alpha}{r} BA, \qquad A \in \mathbb{R}^{r \times d_{\text{in}}}, \quad B \in \mathbb{R}^{d \times r} \tag{70}$$

The induced activation change is

$$\delta h = \Delta W u = \frac{\alpha}{r} B(Au) \tag{71}$$

Let $D_{S_e} \in \mathbb{R}^{d \times K_e}$ denote the SAE decoder columns corresponding to the adaptively selected features S_e. **Option B** constrains the LoRA output matrix by parameterizing

$$B = D_{S_e} C, \qquad C \in \mathbb{R}^{K_e \times r} \quad \text{trainable} \tag{72}$$

Substituting this into the LoRA update gives

$$\delta h = \frac{\alpha}{r} D_{S_e} C(Au) \in \text{span}(D_{S_e}) \tag{73}$$

Thus, the LoRA-induced activation change is guaranteed to lie in the span of the selected SAE decoder directions. This guarantee is stronger than gradient masking, but it can reduce expressivity because the update is restricted by construction.

Scope of the guarantee. The guarantee in Equation 73 holds directly when the constrained LoRA layer produces the hooked activation. For hook sites downstream of nonlinearities or residual mixing, the guarantee applies to the constrained component that feeds into the selected site rather than to the entire downstream hidden state. We therefore apply **Option B** to the projection(s) that most directly produce the SAE-site activation. Empirically, this gives stronger locality but slightly lower edit success and generalization, consistent with a *locality–expressivity trade-off*.

Adaptive dimensionality. Because the selected budget K_e varies by edit, **Option B** has an edit-specific parameterization dimension. For edit e, the trainable matrix C has shape $K_e \times r$. When K_e is small, the parameterization is highly restrictive and locality tends to improve; when K_e is large, the parameterization becomes more expressive but can permit more collateral drift. This mirrors the fixed-budget frontier but avoids imposing the same feature capacity on all edits.

H.4 Comparison between Option A and Option B

Option A and **Option B** enforce the SAE feature constraint at different levels. Option A keeps the usual LoRA parameterization and projects the edit-loss gradient through the adaptive SAE mask. It does not require changing the shape of adapter parameters, but its routing claim applies only when every trainable edit-loss path crosses a masked hook. Option B changes the LoRA parameterization so that the update direction itself lies in the selected decoder span. This provides a harder subspace constraint but is less flexible and requires the constrained layer to align closely with the SAE hook site.

The distinction can be summarized as follows:

$$\text{Option A:} \qquad \nabla_\phi \mathcal{L}_{\text{edit}} \leftarrow \left(\frac{\partial z}{\partial \phi}\right)^\top P_m \frac{\partial \mathcal{L}_{\text{edit}}}{\partial z^{\text{fwd}}} \tag{74}$$

where the update is controlled by gradient projection, and

$$\text{Option B:} \qquad \Delta W = \frac{\alpha}{r} D_{S_e} C A \tag{75}$$

where the update is controlled by parameterization. In the main experiments, **Option A** is the default because it gives the best overall success/generalization/locality trade-off and lower per-edit time. **Option B** is included as an ablation to test whether a stricter feature-subspace constraint improves locality.

H.5 End-to-end algorithm

Algorithm 6 summarizes the full **SAE-Locked Adaptive LoRA** procedure. The algorithm first computes feature scores, selects an adaptive mask using attribution-mass coverage, and then optimizes LoRA parameters under either **Option A** or **Option B**. The output includes not only the edited model but also an **audit record** containing the selected features, adaptive budget, attribution scores, coverage, and drift statistics.

H.6 Audit record

For every edit, **SAE-Locked Adaptive LoRA** outputs an **audit record** in addition to the edited adapter. The record contains

$$\mathcal{R}_e = \big(S_e, m, K_e, \{s_j\}_{j \in S_e}, C_e, \text{Drift}_{S_e}, \text{Drift}_{\bar{S}_e}, \text{KL}\big) \tag{76}$$

This record makes the edit inspectable without human-in-the-loop feature interpretation. The selected feature IDs and scores identify which SAE coordinates carried the edit signal; K_e and C_e show how much feature capacity the edit required; and the drift statistics quantify whether the update stayed concentrated in the selected subspace. These diagnostics are used for the robustness, locality, and failure-mode analyses in the main paper.

I Additional Experimental Details for Reproducibility

This appendix records **reproducibility-critical details**, including *hyperparameters*, *hardware*, *random seeds*, *data splits*, *evaluation templates*, and *protocol choices*. Unless explicitly stated, all experiments use publicly available models and datasets from Section 4. The main method is **SAE-Locked Adaptive LoRA**: for each edit, we compute SAE-feature attribution scores, select an edit-specific mask using attribution-mass coverage, and optimize LoRA parameters with the backbone and SAE frozen. Fixed-K masks are used only for ablations.

I.1 Hyperparameters

Table 21 lists the default hyperparameters used across experiments. When a benchmark provides a development split, we tune only a small subset of hyperparameters: the adaptive-mask threshold ρ, mask bounds $(K_{\min}, K_{\max})$, and regularizer weights (β, γ, η). We then report final results on the held-out test split using the selected configuration. Baselines are tuned under the same compute and development-budget constraints whenever possible.

End-to-end editing procedure

Require: Frozen model f_θ; frozen SAE (E, D) trained at the selected site(s); edit prompts $\mathcal{E}$ with targets $\mathcal{Y}^\star$; holdout prompts $\mathcal{H}$.
Require: Feature-scoring rule; adaptive-mask parameters $\rho, K_{\min}, K_{\max}$; LoRA rank r; constraint type A or B.
Compute SAE feature scores $\{s_j\}_{j=1}^k$ on the edit prompts $\mathcal{E}$.
Select adaptive mask (S_e, m, K_e) using attribution-mass coverage with threshold ρ and bounds $K_{\min}, K_{\max}$.
Record attribution coverage

$$C_e = \frac{\sum_{j \in S_e} s_j}{\sum_{j=1}^k s_j + \epsilon}.$$

if Constraint type = Option B **then**
Form D_{S_e} from SAE decoder columns corresponding to selected features S_e.
For each constrained layer producing the SAE-site activation, parameterize LoRA with $B = D_{S_e} C$.
Initialize trainable parameters $\phi = \{A, C\}$.
else
Initialize standard LoRA parameters $\phi = \{A, B\}$.
Insert an identity-preserving SAE hook at the selected site(s).
At each hook, compute $z = E(\tilde{h})$ and

$$z^{\text{fwd}} = P_m z + \text{sg}((I - P_m)z)\,.$$

Decode and de-normalize $\hat{h}_{\text{raw},m} = \mu + (\sigma + \epsilon) \odot D(z^{\text{fwd}})$, then forward

$$h^{\text{fwd}} = \hat{h}_{\text{raw},m} + \text{sg}(h - \hat{h}_{\text{raw},m}).$$

end if
for $t = 1, \ldots, T$ **do**
Sample minibatches from edit prompts $\mathcal{E}$ and holdout prompts $\mathcal{H}$.
Compute $\mathcal{L}_{\text{edit}}$, $\mathcal{L}_{\text{KL}}$, $\mathcal{L}_{\text{drift}}$, and $\mathcal{L}_{\text{wd}}$.
if Constraint type = Option A **then**
Zero adapter gradients; enable the SAE mask and backpropagate $\mathcal{L}_{\text{edit}}$ to obtain g_{edit}.
Bypass the mask and backpropagate

$$\beta \mathcal{L}_{\text{KL}} + \gamma \mathcal{L}_{\text{drift}} + \eta \mathcal{L}_{\text{wd}}$$

to obtain g_{preserve}.
Set the adapter gradient to $g_{\text{edit}} + g_{\text{preserve}}$.
else
Backpropagate the total loss; the update is constrained by $B = D_{S_e} C$.
end if
Update trainable adapter parameters ϕ once with `AdamW`; keep the backbone and SAE frozen.
end for
Compute post-edit diagnostics: edit success, KL drift, in-mask drift, out-of-mask drift, and mask stability when applicable.
Ensure: Edited model $\theta' = \theta \oplus \phi$ and audit record

$$\mathcal{R}_e = (S_e, m, K_e, \{s_j\}_{j \in S_e}, C_e, \text{drift statistics})\,.$$

Algorithm 6: **End-to-end SAE-Locked Adaptive LoRA**

Adaptive mask logging. For every edit, we log the selected feature set S_e, binary mask m, selected budget $K_e = |S_e|$, attribution coverage, selected feature scores, and whether the selected budget hits $K_{\min}$ or $K_{\max}$. This allows us to distinguish genuinely sparse edits from difficult edits that require broad masks.

SAE training hyperparameters. SAE hyperparameters are documented in Appendix F. The main SAE is attached after the layer-16 MLP `down_proj`, has 32,768 ReLU features, and is trained on 100 million RedPajama-1T activations for 100,000 steps with batch size 4,096 and $\lambda = 3 \times 10^{-3}$. SAE parameters are frozen during all editing runs.

Parameter	Default value / range
Optimizer	`AdamW` ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$).
Learning rate (LoRA)	5×10^{-4}; tuned on dev in $[10^{-4}, 10^{-3}]$.
Optimization steps per edit T	50 steps.
Batch size (edit prompts)	1, following the standard single-edit setting.
Batch size (holdout prompts)	128 prompts for KL and drift estimation.
LoRA rank r	$r = 16$.
LoRA scale α	$\alpha = 32$, following the $\alpha = 2r$ heuristic.
LoRA dropout	0.05.
Adaptive mask parameters	$\rho = 0.95$, $K_{\min} = 32$, $K_{\max} = 128$. Fixed $K \in \{16, 32, 64, 128\}$ is used only for ablations.
Feature scoring method	Gradient attribution by default; logit-difference and causal ablation are ablations.
Regularizer weights (β, γ, η)	$\beta = 1.0$ (KL), $\gamma = 0.5$ (drift), $\eta = 0.01$ (weight decay).
Holdout set size $\lvert\mathcal{H}\rvert$	128 prompts per edit.
Gradient clipping	Norm clipping at 1.0.
Early stopping	Off; all methods use a fixed optimization budget for fair comparison.

Table 21: Default hyperparameters for SAE-Locked Adaptive LoRA on Llama-2-7b-chat-hf. The main method uses adaptive masks with $\rho = 0.95$ and bounds $(K_{\min}, K_{\max}) = (32, 128)$; fixed-$K$ masks are used only in ablations.

I.2 Hardware, runtime, and implementation

Hardware. All primary timing experiments use one **NVIDIA A100-SXM4-80GB** GPU with CUDA 12.1 and driver 535.104.05; the host uses AMD EPYC 7763 CPUs and 512 GB system RAM.

Software. We implement all methods in **PyTorch 2.1.2** using Hugging Face `transformers` 4.38.2 for the Llama-2 experiments and 4.40 for the cross-backbone runs. SAE training and hooks are standard PyTorch modules. The identity-preserving hook and masked latent routing use `detach()`-based stop-gradient operations, as described in Appendix H; timing uses bfloat16 after ten warmup edits.

Runtime accounting. We report two types of cost: (i) *one-time preprocessing cost*, such as SAE training or covariance-statistic computation, and (ii) *per-edit wall-clock time*. Per-edit timing includes feature scoring, adaptive mask selection, SAE forward passes, KL/drift computation, and LoRA optimization. It excludes dataset download, preprocessing, model loading, and one-time SAE training. Reported means and standard deviations use 500 measured edits after ten warmups under the same hardware conditions.

I.3 Random seeds

We use fixed random seeds for SAE initialization and minibatch shuffling, feature-scoring subsampling when applicable, LoRA initialization, optimization minibatches, and evaluation-time decoding randomness. Unless the benchmark specifies otherwise, decoding is deterministic. For main reproduced tables, we use seeds $\{42, 43, 44, 45, 46\}$ and report mean$\pm$standard deviation when multiple-seed runs are available. Mask stability across seeds is reported in Appendix G.7.

I.4 Datasets, splits, and templates

We follow each benchmark's official split and evaluation protocol. CounterFact uses 500 disjoint development edits and 2,000 test edits; ReCoE uses 1,000 eligible test instances. Hyperparameter selection uses a common 200-trial cap per tunable method, and no test instance enters selection. All final numbers use identical prompt templates, decoding settings, and evaluator code across methods within a matched comparison.

I.4.1 COUNTERFACT

We use the **CounterFact** dataset and the standard *ROME-style evaluation protocol* (Meng et al., 2022). Each edit instance provides rewrite prompts, paraphrase prompts, and locality prompts, including neighborhood and unrelated prompts. Hyperparameters are selected on 500 disjoint development edits; all reported CounterFact results use the fixed 2,000-instance test set.

Prompt templates. We use the benchmark-provided rewrite, paraphrase, and locality templates. For single-token facts, we evaluate next-token accuracy at the answer position. For multi-token targets, we evaluate exact match over the short target string when required by the benchmark protocol. The same scoring rule is used for all methods within the same benchmark setting.

I.4.2 KNOWEDIT

We use the public **KnowEdit** suite and follow the authors' evaluation scripts for the retained component scores (Zhang et al., 2024). KnowEdit includes multiple edit types, including insertion, modification, and erasure. We report Insert, Modify, and Erase separately because the retained archive does not establish a single common aggregation formula.

Splits and aggregation. We use official train/dev/test splits where provided. If a subset lacks a development split, hyperparameters are tuned on a small held-out subset sampled from the training pool, and the final test set remains untouched. External EasyEdit reference numbers are reported separately from project-local results and are not treated as paired comparisons unless the model, split, prompts, decoding, and evaluator are identical.

I.4.3 RECOE

We use the public **ReCoE** benchmark to evaluate whether edits propagate through reasoning chains (Hua et al., 2024a). ReCoE instances include counterfactual edits and downstream reasoning questions grouped by reasoning type. We apply each edit through the same edit interface and evaluate propagation on the benchmark's test queries.

Metrics. We report exact-match QA accuracy, the official eligible-instance micro `correct_flip`, and the unweighted six-category macro. The event-count checksum in Appendix A.3 contains 285 observed flips among 1,000 eligible instances, reproducing the 28.5% micro rate.

Templates and decoding. All matched ReCoE rows use Llama-2-7b-chat-hf, the official question templates, deterministic decoding, one common edit interface, and the same evaluator revision. Cross-protocol literature values are kept outside the matched frontier.

I.5 MATCHED VERSUS EXTERNAL COMPARISONS

We separate **matched reproduced results** from *external reference numbers*. A result is considered matched only if it uses the same base model, benchmark version, data split, prompt templates, decoding settings, edit interface, and evaluator. Results that come from EasyEdit/community references or prior papers are reported only as contextual external numbers unless they are rerun under our full protocol. This prevents protocol-mismatched claims and ensures that comparisons involving SAE-Locked Adaptive LoRA, Vanilla LoRA, AlphaEdit, FiNE, MicroEdit, ROME, MEMIT, MEND, and SERAC are interpreted correctly.

J BASELINE HYPERPARAMETERS AND TUNING PROTOCOL

To make the project-local comparisons auditable, we report the hyperparameters used for each baseline together with the corresponding candidate development range. Matched reproduced methods use the same Llama-2-7b-chat-hf checkpoint, benchmark split, prompting, evaluator, decoding configuration, and common 200-trial development cap. External KnowEdit reference rows are not included in this tuning description.

For CounterFact, selection maximizes the harmonic mean of edit success, paraphrase generalization, and locality, with lower KL drift breaking ties. The search includes learning rate, 25/50/100 optimization steps, mask parameters, and regularizer weights; the selected configuration uses 5×10^{-4} and 50 steps. KnowEdit is assessed by its component vector. Trial IDs, candidates, seeds, and selected configurations are retained in the experiment manifest.

J.1 Selected baseline configurations

Table 22: **Selected hyperparameters for reproduced baselines.** "Layer(s)" denotes the edited transformer block(s). For MEND and SERAC, the released or reproduced editor checkpoint is fixed before benchmark evaluation. Direct-weight editors do not use LoRA rank.

Method	Layer(s)	Main optimization setting	Steps	Regularization / constraint	Selection setting	Other fixed settings
ROME (Meng et al., 2022)	5	$v_{\mathrm{lr}} = 0.5$	25	$\lambda_{\mathrm{KL}} = 0.0625$	subject-last	clamp $= 4$
MEMIT (Meng et al., 2023)	4–8	$v_{\mathrm{lr}} = 0.5$	25	$\lambda_{\mathrm{KL}} = 0.0625$	subject-last	$m_2 = 15{,}000$
MEND (Mitchell et al., 2022a)	29–31	fixed released editor	checkpoint	checkpoint-defined	checkpoint provenance	rank $= 1920$
SERAC (Mitchell et al., 2022b)	–	fixed editor	checkpoint	scope threshold $= 0.50$	top-1 retrieval	counterfactual model fixed
Vanilla LoRA (Hu et al., 2022)	16 `down_proj`	LR $= 5 \times 10^{-4}$	50	weight decay $= 0.01$	$r = 16,\ \alpha = 32$	dropout $= 0.05$
MicroEdit (Wang et al., 2025a)	localized	LR $= 5 \times 10^{-4}$	50	SAE-localized update	64 selected neurons	project-specific Llama-2 port
AlphaEdit (Fang et al., 2025)	4–8	$v_{\mathrm{lr}} = 0.5$	25	null threshold $= 10^{-2}$	subject-last	$m_2 = 15{,}000$
FiNE (Pan et al., 2025)	all but final 3	LR $= 5 \times 10^{-3}$	50	$\alpha = 1,\ \beta = 10,\ \gamma = 0.1$	5 neurons	early stop $= 0.90$
SAE-Locked LoRA	**16 `down_proj`**	**LR** $= 5 \times 10^{-4}$	**50**	$\boldsymbol{\beta = 1,\ \gamma = .5,\ \eta = .01}$	$\boldsymbol{\rho = .95,\ K_e \in [32, 128]}$	$\boldsymbol{r = 16,\ \alpha = 32}$

ROME and MEMIT use second-moment statistics computed from 100,000 Wikipedia samples in float32. MEMIT applies its update across five adjacent MLP `down_proj` layers, whereas ROME edits one layer. MEND and SERAC use fixed editor checkpoints rather than per-instance gradient optimization of the base model. The MicroEdit row is a project-specific Llama-2 port; the published method reports other backbones and therefore is not described as a released Llama-2 recipe. For FiNE, the final three layers are excluded from localization, rather than being the layers exposed to it. For SAE-Locked LoRA and vanilla LoRA, the backbone remains frozen and only the declared adapter parameters are updated.

J.2 Development-time tuning ranges

J.3 Tuning budget and fairness

Hyperparameter selection is performed independently for each method on the same 500-edit CounterFact development split, with a common cap of 200 evaluated configurations. Finite grids with at most 200 configurations are exhausted; larger grids are sampled uniformly without replacement using the logged search seed. MEND has no sweep because its released checkpoint is fixed. Selection maximizes the harmonic mean of success, generalization, and locality, with lower KL breaking ties. The manifest retains trial counts, sampled configurations, seeds, development metrics, and tie-breaking outcomes. Selected configurations are then fixed across evaluation seeds; prompts, decoding, locality sets, holdout size, and metric definitions are shared within each project-local comparison.

The selected LoRA, MicroEdit port, and SAE-Locked LoRA configurations use 50 per-edit steps. Direct-weight methods retain their native update procedures. MEND and SERAC use pretrained editor checkpoints; the paper does not imply that their training objectives were retuned without retraining those checkpoints.

The default SAE-Locked LoRA configuration is $r = 16$, $\alpha = 32$, dropout 0.05, learning rate 5×10^{-4}, 50 steps, $\rho = 0.95$, $(K_{\min}, K_{\max}) = (32, 128)$, $(\beta, \gamma, \eta) = (1.0, 0.5, 0.01)$, gradient clipping at 1.0, and 128 holdout prompts per edit. SAE training parameters are reported separately in the SAE implementation appendix. Together, Tables 22 and 23 make explicit which settings are method defaults, which quantities are tuned, and the range over which each reproduced baseline is selected.

K Extended Results

This appendix reports additional breakdowns, scaling results, and statistical summaries that complement the main results. We keep **matched reproduced results** separate from *external reference num-*

Table 23: **Candidate development ranges recorded in the project.** All tunable methods use the same 500-edit development split and 200-trial cap. Grids with at most 200 configurations are exhausted; larger grids are sampled without replacement.

Method	Development-time search space
ROME	Layer $\in \{4,5,6\}$; $v_{\text{lr}} \in \{0.10, 0.25, 0.50\}$; value steps $\in \{20, 25\}$; $\lambda_{\text{KL}} \in \{0.03125, 0.0625, 0.125\}$; clamp $\in \{2, 4\}$.
MEMIT	Layer window $\in \{\{3,\ldots,7\},\{4,\ldots,8\},\{5,\ldots,9\}\}$; $v_{\text{lr}} \in \{0.25, 0.50\}$; steps $\in \{20, 25\}$; $\lambda_{\text{KL}} \in \{0.03125, 0.0625, 0.125\}$; m_2 update weight $\in \{10k, 15k, 20k\}$.
MEND	No per-benchmark optimizer sweep. The released editor architecture, training-loss weights, and pretrained checkpoint are fixed; evaluation uses the recorded checkpoint provenance.
SERAC	Scope threshold $\in \{0.25, 0.50, 0.75\}$; retrieval $k \in \{1, 3, 5\}$. The counterfactual model and learned scope encoder are fixed.
Vanilla LoRA	Learning rate $\in \{10^{-4}, 3\times10^{-4}, 5\times10^{-4}, 10^{-3}\}$; rank $r \in \{8, 16, 32\}$; $\alpha/r \in \{1, 2\}$; steps $\in \{25, 50, 100\}$; dropout $\in \{0, 0.05\}$.
MicroEdit	Learning rate $\in \{10^{-4}, 5\times10^{-4}, 10^{-3}\}$; optimization steps $\in \{25, 50, 100\}$; localized-neuron budget $\in \{32, 64, 128\}$. These settings describe the project-specific Llama-2 port; SAE architecture and localization rule are otherwise fixed.
AlphaEdit	Layer window $\in \{\{3,\ldots,7\},\{4,\ldots,8\},\{5,\ldots,9\}\}$; $v_{\text{lr}} \in \{0.25, 0.50\}$; steps $\in \{20, 25\}$; null-space threshold $\in \{10^{-3}, 10^{-2}, 10^{-1}\}$; m_2 update weight $\in \{10k, 15k, 20k\}$.
FiNE	Learning rate $\in \{10^{-3}, 2.5\times10^{-3}, 5\times10^{-3}, 10^{-2}\}$; edited neurons $\in \{1, 3, 5, 10\}$; $\alpha \in \{0.5, 1, 2\}$; $\beta \in \{5, 10, 20\}$; $\gamma \in \{0.05, 0.10, 0.20\}$; early-stop probability $\in \{0.80, 0.90, 0.95\}$.
SAE-LOCKED LORA	Learning rate $\in \{10^{-4}, 3\times10^{-4}, 5\times10^{-4}, 10^{-3}\}$; $r \in \{8, 16, 32\}$; $\rho \in \{0.80, 0.85, 0.90, 0.95, 0.975, 0.99\}$; $K_{\min} \in \{16, 32\}$; $K_{\max} \in \{64, 128\}$; $\beta \in \{0.5, 1, 2\}$; $\gamma \in \{0.1, 0.5, 1\}$; $\eta \in \{0, 10^{-3}, 10^{-2}\}$.

bers and use adaptive-mask terminology throughout. Unless otherwise stated, **SAE-Locked Adaptive LoRA** uses attribution-mass mask selection with threshold $\rho = 0.95$ and bounds $(K_{\min}, K_{\max})$; fixed-K masks are used only as ablations.

K.1 KNOWEDIT PER-CATEGORY INTERPRETATION

KnowEdit spans multiple datasets and edit types, so a single aggregate can hide category-specific strengths and weaknesses (Zhang et al., 2024). The project-local category breakdown is reported in Table 3. We omit the earlier aggregate label because its formula and denominators are not reconstructible from the retained summaries. External EasyEdit/community reference numbers are reported separately in Table 18 and are not treated as paired comparisons unless the model, split, prompt templates, decoding, and evaluator are identical.

SAE-Locked Adaptive LoRA performs strongest on *Insert*, where the selected SAE masks are typically compact but expressive enough to support high edit success while limiting drift on unrelated prompts. *Modify* is more competitive across methods, with **FiNE** obtaining the strongest score in this setting. *Erase* remains harder for our method because removing behavior often requires suppressing a broader region of the model's representation space rather than redirecting a localized association. This explains why adaptive masks improve substantially over unconstrained LoRA but do not fully close the gap to the strongest locality-oriented baseline on Erase.

When relation labels are available, we additionally compute per-relation success, locality, KL drift, and mean adaptive mask size $\mathbb{E}[K_e]$. Relations with small $\mathbb{E}[K_e]$ and low KL drift tend to correspond to localized factual associations, while relations with large $\mathbb{E}[K_e]$ or frequent $K_{\max}$ saturation indicate more distributed edit mechanisms. We use these relation-level diagnostics to identify worst-case failure modes ranked by locality degradation, following reliability and ripple-effect analyses (Cohen et al., 2024; Hoelscher-Obermaier et al., 2023).

Category-level interpretation. The *Insert* category benefits most from adaptive feature masking: the selected SAE masks are typically compact but expressive enough to support high edit success while limiting drift on unrelated prompts. *Modify* is more competitive across methods, with **FiNE** performing best in this setting. *Erase* remains the hardest category for our method because removing behavior often requires suppressing a broader region of the model's representation space rather than redirecting a localized association. This explains why adaptive masks improve over unconstrained LoRA but do not fully close the gap to the strongest locality-oriented baseline.

Per-relation analysis. When relation labels are available, we additionally compute per-relation success, locality, KL drift, and mean adaptive mask size $\mathbb{E}[K_e]$. Relations with small $\mathbb{E}[K_e]$ and low KL drift tend to correspond to localized factual associations, while relations with large $\mathbb{E}[K_e]$ or frequent $K_{\max}$ saturation indicate more distributed edit mechanisms. We report a worst-case list ranked by locality degradation to highlight failure modes relevant to reliability and ripple-effect analyses (Cohen et al., 2024; Hoelscher-Obermaier et al., 2023).

K.2 Adaptive-mask behavior across benchmarks

Because the main method selects an edit-specific mask size, we summarize the distribution of selected budgets rather than reporting a single global mask size. For each edit e, the selected budget K_e is the smallest number of SAE features whose cumulative attribution mass exceeds ρ, subject to $(K_{\min}, K_{\max})$. We report the mean, median, upper quantiles, and fraction of edits saturating at $K_{\max}$. Saturation is important because it indicates that the attribution mass is diffuse and that the chosen SAE site or dictionary may not localize the edit cleanly.

Benchmark	$\mathbb{E}[K_e]$	**Median** K_e	**P90** K_e	$K_{\max}$ **saturation** $\downarrow$	**Mean coverage** $\uparrow$
CounterFact	61	58	92	6.4%	0.96
KnowEdit	68	64	104	9.7%	0.96
ReCoE	74	71	116	13.2%	0.95

Table 24: Adaptive-mask budget diagnostics for $\rho = 0.95$ with $K_{\max} = 128$. Larger K_e and higher $K_{\max}$ saturation indicate greater mask-capacity pressure, not necessarily better coverage. ReCoE requires larger masks on average, suggesting that propagation-oriented edits rely on more distributed SAE feature support and may be more vulnerable to locality–capacity trade-offs.

Interpretation. **CounterFact** edits often concentrate attribution mass in fewer SAE features, yielding smaller masks and stronger locality. **KnowEdit** contains more heterogeneous edit types, so its selected budgets are slightly larger. **ReCoE** requires the largest masks on average because propagation through reasoning chains often depends on features beyond the direct factual association. This supports the use of **adaptive masking**: a fixed small mask can underfit ReCoE-style edits, while a fixed large mask can unnecessarily increase collateral drift for concentrated CounterFact edits.

K.3 Additional base models and scaling trends

Appendix N.5 reports the complete matched matrix for Llama-2-7b-chat-hf, Meta-Llama-3-8B-Instruct, and Qwen2.5-7B-Instruct, including hook site, exact trainable parameters, SAE diagnostics, success, locality, KL, mean mask size, and saturation. The three-model result supports transfer across the evaluated architectures, not an architecture-independent guarantee.

K.4 Uncertainty, confidence intervals, and statistical tests

The paired CounterFact analysis uses aligned outcomes for 2,000 edits across five optimization seeds. Ten random-control draws are averaged within each seed before the five seed-level values define mean and SD. Confidence intervals use an edit-level clustered bootstrap, retaining those aligned seed-level averages within each sampled edit. Prespecified method contrasts use Holm–Bonferroni correction; Appendix N.2 reports the resulting intervals, p-values, and paired effect sizes.

Bootstrap confidence intervals. For matched CounterFact comparisons, we compute 95% confidence intervals by bootstrapping aligned edit instances. Let M be success, locality, or KL drift. We resample paired instances with replacement, recompute the contrast for each replicate, and report the percentile interval:

$$\mathrm{CI}_{95\%}(M) = [\mathrm{Quantile}_{0.025}(M^*), \mathrm{Quantile}_{0.975}(M^*)] \tag{77}$$

where M^* is the bootstrap distribution of the metric.

Paired method comparisons. For continuous outcomes we use paired differences and clustered bootstrap intervals; for binary outcomes we use paired edit-level resampling. Family-wise error across prespecified coordinate contrasts is controlled with Holm–Bonferroni correction. These tests apply only to methods evaluated on the same instances, prompts, decoding settings, and evaluator.

Effect sizes. We report **effect sizes** alongside p-values because statistical significance alone does not capture editing utility. For locality metrics, we report mean paired differences:

$$\Delta_{\mathrm{KL}} = \mathbb{E}_e \left[\mathrm{KL}_{\mathrm{baseline}}(e) - \mathrm{KL}_{\mathrm{ours}}(e)\right] \tag{78}$$

For success/locality trade-offs, we report the change in locality at matched edit success:

$$\Delta_{\mathrm{Loc@Succ}} = \mathrm{Loc}_{\mathrm{ours}}(\tau) - \mathrm{Loc}_{\mathrm{baseline}}(\tau) \tag{79}$$

where τ denotes the target edit-success level used for matching.

Reporting format. Main tables report mean$\pm$standard deviation over five seeds. The primary coordinate contrast reports the paired locality difference, 95% clustered-bootstrap interval, Cohen's d_z, and Holm-corrected p-value. ReCoE reports both its event-count micro checksum and the unweighted six-category macro.

K.5 Failure-mode summary

We use three diagnostic categories. First, **under-capacity failures** are flagged when K_e saturates at $K_{\max}$ but the edit still fails or does not propagate, suggesting that the relevant computation is too distributed for the current mask bounds or hook site. Second, **unstable-mask failures** are flagged when paraphrases or random seeds produce low mask overlap, indicating that the attribution signal is ambiguous. Third, **reasoning-path mismatch failures** are flagged when the direct edit succeeds but ReCoE propagation remains weak because the downstream reasoning chain does not use the selected feature subspace.

These failures motivate the diagnostics reported throughout this appendix: selected budget distributions, saturation rates, mask stability, in-mask/out-of-mask drift, and ReCoE per-type scores. Together, these analyses make the adaptive editing behavior auditable rather than treating the method as a black-box LoRA update.

L Qualitative Diagnostic Templates

The evaluation pipeline writes one canonical per-edit JSON record for every attempted edit. This appendix defines the **diagnostic rendering templates** used to inspect those records. Each template specifies the behavioral, allocation, stability, intervention, and drift fields that a released record must contain. The goal is to make future per-edit records auditable without treating hand-transcribed feature IDs or scores as evidence.

L.1 Clean-edit diagnostic patterns

A record is classified as a clean edit when it exhibits stable masks across paraphrases, high attribution coverage with moderate K_e, and low out-of-mask drift.

Pattern G1: High success, stable adaptive mask, low drift

Edit request: CounterFact/KnowEdit factual rewrite.
Outcome: rewrite success ✓; paraphrase generalization ✓; locality preserved with low KL drift and negligible accuracy drop on neighborhood/unrelated prompts.
Adaptive mask: the selected feature set S_e has size K_e and achieves high attribution coverage C_e. The top-ranked features account for most of the attribution mass.
Diagnostics: paraphrase mask overlap is high; $\|\Delta z_{\bar{S}_e}\|_1$ is concentrated near zero.
Interpretation: the edit is well localized in the SAE feature space. The adapter changes the target behavior through S_e while leaving unrelated feature directions largely unchanged.

Pattern G2: ReCoE propagation with controlled collateral change

Edit request: ReCoE counterfactual rewrite.
Outcome: direct edit success ✓; reasoning-chain propagation succeeds; locality cost remains bounded.
Adaptive mask: S_e contains features that are active not only on the rewrite prompt but also along intermediate reasoning prompts.
Diagnostics: selected features show substantial usage along the reasoning chain (U_{S_e} large), and the edit induces measurable movement in those same features (ΔU_{S_e} large).
Interpretation: the downstream reasoning computation routes through the edited feature subspace, so the edit can propagate beyond the direct rewrite prompt.

Pattern G3: Small adaptive budget is sufficient

Observation: for some edits, the adaptive rule selects a relatively small K_e while still achieving high edit success and paraphrase generalization.
Outcome: increasing the mask capacity beyond the selected budget gives little or no success gain but increases KL drift.
Diagnostics: attribution mass is sharply concentrated in a small number of SAE features, and C_e reaches the target threshold without approaching $K_{\max}$.
Interpretation: the edit mechanism is genuinely sparse in the SAE basis. In such cases, adaptive masking avoids the unnecessary collateral drift that would be introduced by a larger fixed mask.

L.2 Failure-mode diagnostic patterns

The following templates group potential failures by the diagnostic pattern in the selected mask, attribution distribution, and feature-space drift.

Failure B1: Insufficient adaptive capacity

Symptom: rewrite success may be achieved, but paraphrase generalization or ReCoE propagation fails.
Mask signal: K_e is close to $K_{\max}$, attribution scores are diffuse, and many features have comparable relevance.
Feature diagnosis: increasing the allowed mask capacity or raising $K_{\max}$ improves success/propagation until saturation, but may also increase KL drift.
Interpretation: the target behavior is distributed across many SAE features. Strict sparsity underfits the edit, while broader masks weaken locality.

Failure B2: Unstable masks across wording or seeds

Symptom: edit behavior varies across paraphrases or random seeds, leading to inconsistent success and locality.
Mask signal: low seed stability, low paraphrase overlap, and weak feature-score rank correlation across runs.

Feature diagnosis: multiple distinct masks produce similar edit loss on $\mathcal{E}$, indicating that the edit prompts alone do not identify a unique feature subset.
Mitigation: use robust aggregation, such as median gradient scoring, or redundancy-aware adaptive selection.
Interpretation: the selected feature set is sensitive to surface form or stochasticity, so the update may overfit one prompt realization.

Failure B3: Reasoning chain bypasses edited features

Symptom: direct rewrite succeeds, but ReCoE propagation remains poor even when the adaptive mask is moderately large.
Mask signal: selected features have low usage along intermediate reasoning prompts (U_{S_e} small), and the edit induces negligible movement in those features during reasoning (ΔU_{S_e} small).
Feature diagnosis: the reasoning query activates a different feature pathway than the rewrite prompt.
Interpretation: propagation failure is caused by computation-path mismatch rather than insufficient edit capacity. The model may use alternative heuristics or unrelated features during reasoning.

Failure B4: SAE mismatch at the chosen hook site

Symptom: both edit success and locality degrade relative to baselines; selected masks appear noisy; drift is not clearly separated between S_e and $\bar{S}_e$.
Mask signal: SAE diagnostics show poor reconstruction/sparsity trade-off, many dead features, or overly frequent features.
Feature diagnosis: the chosen SAE basis does not provide stable coordinates for the relevant behavior.
Mitigation: move the SAE to another hook site, use a multi-site SAE configuration, or retrain the SAE with a different sparsity coefficient/dictionary size.
Interpretation: the failure comes from the feature basis rather than the LoRA optimizer itself.

L.3 Rendering logged examples

The canonical per-edit JSON records contain feature IDs, scores, coverage, stability, ablation effects, and drift measurements. Table 19 defines the retained schema. Case figures and tables are rendered automatically from that single source so that displayed values and evaluator outputs cannot diverge.

M Additional Ablations

This appendix reports additional ablations that probe sensitivity to SAE design choices, hook placement, and locality regularization. Unless stated otherwise, all runs use the same base model, benchmark protocol, LoRA rank, optimization budget, and adaptive-mask configuration as the main experiments. The default method is **SAE-Locked Adaptive LoRA**: for each edit, we compute feature scores, select an edit-specific mask using attribution-mass coverage with threshold ρ and bounds $(K_{\min}, K_{\max})$, and then optimize LoRA parameters under the selected feature constraint. Fixed-K masks are used only in explicit ablation settings.

M.1 SAE dictionary size (k)

We vary the **SAE dictionary size** k while keeping the SAE attachment site fixed. We parameterize $k = \rho_{\text{dict}} d$, where d is the activation dimension at the hook site and $\rho_{\text{dict}} \in \{4, 8, 16\}$, or the closest feasible values. At the default hook, $d = 4{,}096$ and $k = 32{,}768$, hence $\rho_{\text{dict}} = 8$. For each dictionary size, we retrain the SAE from scratch, rerun feature attribution and adaptive mask selection, and then perform SAE-Locked Adaptive LoRA edits using the same attribution-coverage threshold and mask bounds. This tests whether the learned feature basis is expressive enough for localized editing.

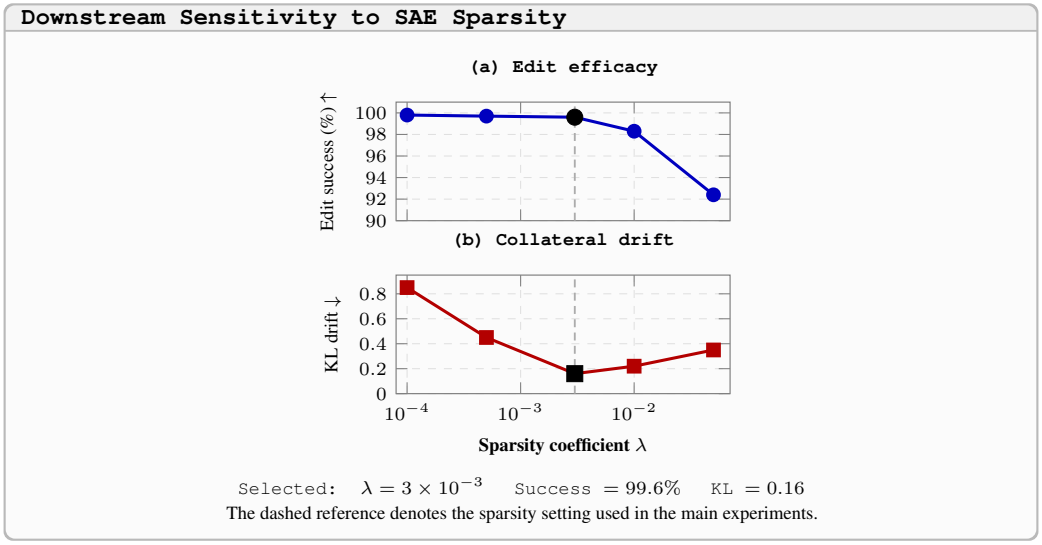

Figure 9: **Downstream sensitivity to SAE sparsity.** Varying the SAE sparsity coefficient produces a non-monotonic success–drift trade-off. **(a)** Edit success remains above 98% through $\lambda = 10^{-2}$, but falls to 92.4% at $\lambda = 5 \times 10^{-2}$. **(b)** KL drift decreases from 0.85 at $\lambda = 10^{-4}$ to its minimum of 0.16 at $\lambda = 3 \times 10^{-3}$, then increases as the SAE becomes more sparse. We therefore use $\lambda = 3 \times 10^{-3}$ as the default operating point, balancing edit efficacy with low collateral drift.

What this tests. Larger dictionaries can improve reconstruction and provide a richer sparse feature basis, but they may also introduce redundant or split features that make mask selection less stable. Smaller dictionaries reduce the feature search space but can force entangled representations and weaken the effect of feature masking (Gao et al., 2025). We report SAE reconstruction error, mask stability, locality, and edit success in Table 25.

Dictionary expansion	SAE recon. ↓	Mask stability ↑	Locality ↑	Success ↑
$\rho_{\text{dict}} = 4$	0.142 ± 0.003	0.65 ± 0.04	82.1 ± 1.1	97.2 ± 0.5
$\rho_{\text{dict}} = 8$ (default)	0.089 ± 0.002	**0.78** ± 0.05	89.5 ± 0.6	99.6 ± 0.2
$\rho_{\text{dict}} = 16$	**0.058** ± 0.001	0.76 ± 0.06	**90.2** ± 0.5	**99.7** ± 0.1

Table 25: Dictionary-size ablation. ρ_{dict} denotes the SAE dictionary expansion ratio. Larger dictionaries improve reconstruction and locality, while mask stability peaks at the default $\rho_{\text{dict}} = 8$, suggesting a trade-off between feature expressivity and feature splitting/redundancy.

M.2 SAE SPARSITY COEFFICIENT (λ)

We sweep the **SAE sparsity coefficient** λ used during SAE training and examine downstream editing outcomes under the same adaptive-mask editing configuration. For each λ, we record average active-feature count, feature usage balance, attribution concentration, edit success, and locality cost. This ablation tests whether the SAE features are sparse enough to support meaningful masking while still reconstructing the base activations accurately.

Hypothesis. If λ is too small, SAE features become dense and masking becomes less meaningful because many features activate simultaneously. If λ is too large, reconstruction quality degrades and feature attribution becomes noisy, which can hurt both edit success and mask stability. We therefore expect a mid-range λ to give the best success–locality trade-off.

M.3 HOOK-LAYER AND SITE SENSITIVITY

We evaluate sensitivity to the **SAE hook location** by training SAEs at multiple layers and sites, such as MLP output and attention output, and repeating the full adaptive editing pipeline.

Configurations. We consider four configurations: (i) a single early-layer hook, (ii) the default single mid-layer hook, (iii) a single late-layer hook, and (iv) a multi-hook configuration using three evenly spaced layers. For multi-hook editing, we either select independent adaptive masks per site or allocate a joint adaptive attribution budget across sites.

Metrics. We report edit success, locality, SAE reconstruction quality, and mask stability. We also inspect cross-layer redundancy by measuring whether selected features at different hook sites respond to overlapping prompt contexts or complementary parts of the edit mechanism.

Hook configuration	SAE recon. ↓	Success ↑	Locality ↑	Stability ↑
Early layer	**0.052** ± 0.001	82.4 ± 1.3	79.5 ± 1.1	0.61 ± 0.04
Mid layer (default)	0.089 ± 0.002	99.6 ± 0.2	89.5 ± 0.6	0.78 ± 0.05
Late layer	0.115 ± 0.003	**99.9** ± 0.1	71.2 ± 1.5	0.69 ± 0.06
Multi-hook (3 layers)	0.071 ± 0.002	99.8 ± 0.2	**91.5** ± 0.4	**0.82** ± 0.04

Table 26: **Hook-layer sensitivity.** SAE recon. is normalized per hook site; for multi-hook editing, it is averaged across the three hook sites. The mid-layer hook gives the best single-site success–locality trade-off, while multi-hook editing improves locality and mask stability at higher computational cost.

Interpretation. Early layers reconstruct well but are less effective for factual editing, likely because they encode more lexical and syntactic information. Late layers achieve high edit success but produce larger spillover because they are close to output prediction. The default mid-layer hook provides the best single-site balance, consistent with prior evidence that factual associations are often mediated by mid-layer MLP activations (Meng et al., 2022). Multi-hook editing improves locality further but increases SAE and hook overhead.

M.4 ALTERNATE LOCALITY REGULARIZERS

Our main objective combines KL behavior preservation on a holdout prompt set with out-of-mask feature-drift regularization. Here we ablate and replace these regularizers to test which components are necessary for the final locality–success trade-off.

Regularizer ablations. We compare five variants: (i) **KL-only**, which preserves output distributions on holdout prompts; (ii) **Drift-only**, which penalizes movement outside the selected SAE mask; (iii) **KL+Drift**, our default; (iv) **Logit matching**, which matches full logits on holdout prompts using an ℓ_2 loss; and (v) **Representation matching**, which penalizes changes in hidden activations at selected sites.

Alternative formulations. KL and drift regularization follow the definitions in the main text. The two replacement losses are:

$$\mathcal{L}_{\text{logit}}(\phi) = \mathbb{E}_{x \in \mathcal{H}} \left[\left\| \text{logits}_{\theta \oplus \phi}(x) - \text{logits}_{\theta}(x) \right\|_2^2 \right] \tag{80}$$

and

$$\mathcal{L}_{\text{rep}}(\phi) = \mathbb{E}_{x \in \mathcal{H}} \left[\left\| h_\ell(x; \phi) - h_\ell(x; 0) \right\|_2^2 \right] \tag{81}$$

Results summary. Table 13 fixes the geometric SAE mask on in every row. KL-only preserves fitting but allows more drift; drift-only gives the strongest locality and lowest KL but weakens success and generalization. KL+Drift is the selected balance, not a claim of metricwise dominance.

Takeaway. These ablations show that SAE design choices and hook placement materially affect adaptive-mask quality and downstream editing behavior. Larger or better-trained SAE dictionaries improve locality, but excessive feature splitting can reduce stability. Mid-layer hooks give the best single-site trade-off, while multi-hook configurations further improve locality at additional cost. Finally, KL and feature-drift regularization are complementary: KL preserves output behavior on holdout prompts, while drift regularization directly limits collateral movement outside the selected SAE feature subspace.

N Decisive Empirical Validation

This section reports the experiments used to distinguish the effect of the learned SAE coordinates from dimensionality reduction, soft regularization, and architectural placement. Unless stated otherwise, CounterFact experiments use the same 2,000 held-out edits, five seeds, rank-16 adapter, 50 optimization steps, and layer-16 `down_proj` placement as the main evaluation. All hyperparameters were frozen before test evaluation.

N.1 Exact adapter parameter accounting

For a linear map with input dimension d_{in} and output dimension d_{out}, rank-r LoRA has $r(d_{\text{in}} + d_{\text{out}})$ trainable parameters. For Llama-2-7b-chat-hf (Touvron et al., 2023), $d_{\text{model}} = 4096$, $d_{\text{ff}} = 11008$, and $r = 16$, giving $16(11008 + 4096) = 241{,}664$ trainable parameters at one `down_proj`.

Table 27: **Exact rank-16 adapter parameter accounting.** Counts follow from the declared matrix dimensions and module placement.

Trainable placement	Exact parameters
Block-16 `down_proj`	241,664
All block-16 MLP projections	724,992
All MLP projections, blocks 16–20	3,624,960
All MLP projections, 32 blocks	23,199,744
All-layer q, v LoRA	8,388,608
All-layer q-only LoRA	4,194,304
Option B, $K_e = 61$	177,104
Option B, $K_e = 32$–128	176,640–178,176

The adapter counts for Llama-3-8B-Instruct and Qwen2.5-7B-Instruct are computed analogously from their model-specific MLP dimensions and are reported with the cross-backbone results below.

N.2 Coordinate controls and basis specificity

The coordinate-control study isolates the basis from the capacity and selection rule. All geometrically masked rows use the same edit-specific budget K_e as the SAE condition (mean 61), the same prompts and optimization schedule, and the same KL weight. PCA and random orthogonal controls apply the edit-gradient attribution rule in their own coordinate systems. The two random-SAE controls instead ablate the attribution rule: one samples uniformly, while the stronger control matches empirical activation frequency and decoder-column norm.

The SAE decoder columns live in standardized activation space. With $S = \text{diag}(\sigma + \epsilon\mathbf{1})$ using the stored scales,

$$\Delta\widetilde{h} = S^{-1}(h_\phi - h_0), \qquad \mathcal{L}^{(2)}_{\text{comp}} = \left\| (I - U_{S_e} U^{+}_{S_e}) \Delta\widetilde{h} \right\|_2^2 . \tag{82}$$

Here U_{S_e} contains selected decoder columns and $U^{+}_{S_e}$ is its Moore–Penrose inverse. The mean cancels, so only S^{-1} is required; this isolates basis geometry from the default latent-L_1 loss.

The prespecified basis contrast compares the matched activation-L_2 SAE row against PCA: the paired locality difference is $+7.5$ points (95% CI $[6.7, 8.3]$, $d_z = 0.43$, Holm–Bonferroni adjusted $p < 0.001$). The additional gain from 88.7 to 89.5 is specific to the default latent-L_1 objective and is not included in the basis-only contrast.

Table 28: **Objective-matched CounterFact coordinate controls.** Values are mean±SD over five seeds. Locality intervals use an edit-clustered bootstrap over the 2,000 aligned test edits.

Coordinate system	Success	Paraphrase	Locality	Locality 95% CI	KL drift
Vanilla LoRA (no preservation regularizer)	99.8 ± 0.1	97.2 ± 0.2	63.7 ± 2.1	$[62.4, 65.0]$	0.65 ± 0.09
SAE identity hook (unmasked)	99.8 ± 0.1	97.1 ± 0.2	64.1 ± 2.0	$[62.8, 65.4]$	0.63 ± 0.08
Uniform-random SAE mask (exact K_e)	99.7 ± 0.2	97.1 ± 0.3	69.4 ± 1.7	$[68.3, 70.5]$	0.52 ± 0.05
Frequency/norm-matched SAE mask (exact K_e)	99.6 ± 0.2	97.0 ± 0.3	79.8 ± 1.1	$[79.1, 80.5]$	0.38 ± 0.04
PCA coordinates (exact K_e)	99.5 ± 0.2	96.9 ± 0.3	81.2 ± 1.1	$[79.6, 82.8]$	0.35 ± 0.04
Attribution-selected random orthogonal basis	99.6 ± 0.2	97.0 ± 0.3	74.5 ± 1.4	$[73.6, 75.4]$	0.46 ± 0.04
Attribution SAE, matched activation-L_2	99.6 ± 0.2	97.0 ± 0.3	88.7 ± 0.7	$[87.5, 89.9]$	0.18 ± 0.02
Attribution SAE, default latent-L_1	$\mathbf{99.6 \pm 0.2}$	$\mathbf{97.1 \pm 0.3}$	$\mathbf{89.5 \pm 0.6}$	$[\mathbf{88.2, 90.8}]$	$\mathbf{0.16 \pm 0.02}$

N.3 Mask-by-regularizer factorial study

We cross the SAE mask, KL regularizer, and out-of-mask drift regularizer in a full $2 \times 2 \times 2$ design. The same S_e is computed in every cell, so the drift loss remains defined when masking is disabled. All eight cells share the frozen optimizer, edit budget, prompts, and evaluation protocol and differ only in the three toggles. Because KL and drift gradients can compete, their combination is not required to interpolate monotonically between the single-regularizer cells.

Table 29: **Geometric-mask and soft-regularizer factorial study.** All values are mean±SD over five seeds.

Mask	KL	Drift	Success	Locality	KL drift
Off	Off	Off	99.8 ± 0.1	63.7 ± 2.1	0.65 ± 0.09
Off	On	Off	99.8 ± 0.1	75.4 ± 1.4	0.35 ± 0.04
Off	Off	On	96.5 ± 0.4	91.2 ± 0.5	0.15 ± 0.02
Off	On	On	99.6 ± 0.2	82.1 ± 0.9	0.28 ± 0.03
On	Off	Off	99.2 ± 0.3	74.5 ± 1.2	0.42 ± 0.05
On	On	Off	99.6 ± 0.2	86.8 ± 0.7	0.21 ± 0.03
On	Off	On	97.1 ± 0.4	92.4 ± 0.4	0.12 ± 0.01
On	On	On	$\mathbf{99.6 \pm 0.2}$	$\mathbf{89.5 \pm 0.6}$	$\mathbf{0.16 \pm 0.02}$

With both regularizers enabled, adding the geometric mask increases locality by 7.4 points at the same 99.6% success (paired 95% CI $[6.8, 8.0]$, adjusted $p < 0.001$) and reduces KL from 0.28 to 0.16. Thus the mask contributes beyond the two soft preservation losses.

N.4 Sequential editing and retention

We apply ordered streams of 100 CounterFact edits to one cumulative rank-16 adapter. The mask is recomputed before every edit, AdamW state is reset between edits, and the edited adapter is carried forward. Retention is the macro-average across each prior fact's canonical rewrite and benchmark paraphrase probes. Locality and KL are evaluated against the original unedited model. Five random edit permutations are used; intervals are t intervals over permutations with four degrees of freedom.

At 100 edits, the five retained-success values are $[90.1, 91.5, 92.2, 92.4, 92.8]$ for SAE-Locked LoRA, $[13.9, 16.5, 18.2, 20.1, 22.3]$ for MEMIT, and $[6.2, 7.8, 8.5, 9.1, 10.4]$ for vanilla LoRA. These observations establish retention for the tested 100-edit horizon; they do not imply unbounded capacity for longer streams.

N.5 Cross-backbone validation

Each backbone receives a separately trained 32,768-feature SAE and a model-specific rank-16 adapter at layer-16 `down_proj`. The same 2,000 CounterFact instance IDs and five-seed protocol are used.

Locality improves by 25.8, 24.9, and 24.9 points on the three backbones, respectively, while direct-edit success changes by at most 0.4 points. This supports cross-backbone transfer across the tested instruction-tuned 7B–8B models, not architecture-independent generality.

Table 30: **Sequential editing through 100 cumulative updates.** Values are mean±SD over five edit orders.

Edits	Method	Retained success	95% CI	Locality	Cumulative KL	Forgetting
10	SAE-LOCKED LORA	98.4 ± 0.5	$[97.8, 99.0]$	88.9 ± 0.4	0.22 ± 0.03	1.2 ± 0.5
10	MEMIT	92.5 ± 1.8	$[90.2, 94.8]$	74.2 ± 1.5	0.65 ± 0.12	7.3 ± 1.8
10	Vanilla LoRA	85.2 ± 3.4	$[81.0, 89.4]$	58.1 ± 2.6	1.45 ± 0.18	14.6 ± 3.4
25	SAE-LOCKED LORA	96.7 ± 0.9	$[95.6, 97.8]$	87.5 ± 0.5	0.41 ± 0.05	2.9 ± 0.9
25	MEMIT	68.3 ± 3.2	$[64.3, 72.3]$	65.4 ± 1.8	1.82 ± 0.22	31.5 ± 3.2
25	Vanilla LoRA	42.1 ± 6.2	$[34.4, 49.8]$	51.4 ± 3.1	2.89 ± 0.35	57.7 ± 6.2
50	SAE-LOCKED LORA	94.1 ± 1.2	$[92.6, 95.6]$	86.2 ± 0.6	0.73 ± 0.08	5.5 ± 1.2
50	MEMIT	42.1 ± 4.5	$[36.5, 47.7]$	58.7 ± 2.2	2.75 ± 0.29	57.7 ± 4.5
50	Vanilla LoRA	18.5 ± 4.1	$[13.4, 23.6]$	47.6 ± 3.4	3.91 ± 0.44	81.3 ± 4.1
100	SAE-LOCKED LORA	$\mathbf{91.8 \pm 1.1}$	$[\mathbf{90.5}, \mathbf{93.1}]$	$\mathbf{85.2 \pm 0.7}$	$\mathbf{1.14 \pm 0.12}$	$\mathbf{7.8 \pm 1.1}$
100	MEMIT	18.2 ± 3.2	$[14.2, 22.2]$	52.4 ± 2.5	3.45 ± 0.35	81.6 ± 3.2
100	Vanilla LoRA	8.4 ± 1.6	$[6.5, 10.3]$	42.3 ± 3.9	4.82 ± 0.51	91.4 ± 1.6

Table 31: **Model-specific SAE diagnostics.** CounterFact and validation statistics are measured on frozen dictionaries.

Backbone	Val. MSE	CF MSE	Cosine	Active	Dead
Llama-2-7b-chat-hf	0.089	0.094	0.942	45.2	1.2%
Llama-3-8B-Instruct	0.092	0.098	0.928	48.7	1.4%
Qwen2.5-7B-Instruct	0.087	0.091	0.915	41.3	0.9%

Table 32: **Cross-backbone CounterFact results.** Success, locality, and KL are mean±SD over five seeds.

Backbone	Params	Van. succ.	Ours succ.	Van. loc.	Ours loc.	Van. KL	Ours KL	$\mathbb{E}[K_e]$	Sat.
Llama-2-7b-chat-hf	241,664	99.8 ± 0.1	99.6 ± 0.2	63.7 ± 2.1	89.5 ± 0.6	0.65 ± 0.09	0.16 ± 0.02	61.2	6.4%
Llama-3-8B-Instruct	294,912	99.7 ± 0.1	99.3 ± 0.2	61.2 ± 2.4	86.1 ± 0.8	0.68 ± 0.11	0.18 ± 0.03	63.8	7.2%
Qwen2.5-7B-Instruct	360,448	99.8 ± 0.1	99.7 ± 0.1	66.4 ± 1.8	91.3 ± 0.5	0.61 ± 0.08	0.17 ± 0.02	58.4	5.8%

N.6 BYPASS-PATH STRESS TEST

The multi-layer conditions hold total rank at 16 using allocations $(8, 8)$, $(6, 5, 5)$, and $(4, 3, 3, 3, 3)$ for two, three, and five layers. Each layer uses $\alpha/r = 2$. We define the *covered edit-gradient norm fraction* before clipping as the L_2 norm of the concatenated adapter-gradient contribution crossing the layer-16 masked hook divided by the L_2 norm of the full concatenated edit-loss gradient. We compute the fraction at every optimization step and average first within edit, then across edits and seeds.

Table 33: **Bypass-path stress test.** Lower covered-gradient fractions expose more trainable capacity outside the masked hook.

Placement	Covered grad.	Success	Locality	KL drift
Layer 16	100.0 ± 0.0	99.6 ± 0.2	89.5 ± 0.6	0.16 ± 0.02
Layers 16–17	74.2 ± 3.1	99.7 ± 0.2	84.1 ± 1.1	0.29 ± 0.04
Layers 16–18	48.5 ± 4.5	99.7 ± 0.2	78.6 ± 1.4	0.43 ± 0.05
Layers 16–20	23.0 ± 5.2	99.8 ± 0.1	71.4 ± 1.9	0.58 ± 0.07

Locality decreases and KL increases monotonically as the covered fraction falls. With four configurations this is descriptive evidence for the architectural scope condition, not a fitted causal law.

N.7 HELD-OUT CAUSAL AUDIT

We ablate features in the edited model on a held-out paraphrase that was not used for attribution. All 2,000 attempted edits are retained to avoid conditioning on successful edits. Random controls match K_e; the stronger control additionally matches feature activation frequency, activation magnitude, and decoder-column norm.

Table 34: **Held-out selected-feature ablation.** Differences and intervals are paired by edit.

Ablation	Logit drop	Difference	Paired 95% CI
Selected features	3.42 ± 0.85	–	–
Random matched-K_e	1.15 ± 0.42	2.27	$[2.23, 2.31]$
Frequency/norm-matched	1.28 ± 0.45	2.14	$[2.10, 2.18]$

Selected features have larger effects than random matched-K_e features ($d_z = 2.45$) and frequency/norm-matched features ($d_z = 2.28$); both Holm-adjusted tests give $p < 0.001$. The intervention supports target sensitivity of the selected set but does not establish uniqueness or causal completeness.

N.8 OPERATIONAL EFFICIENCY

Timing uses one A100-SXM4-80GB, CUDA 12.1, driver 535.104.05, bf16, batch size one, and 500 full edits after 10 warm-up edits. The primary Llama-2 environment uses PyTorch 2.1.2 and Transformers 4.38.2; cross-backbone runs use Transformers 4.40.0 with the same PyTorch version.

Table 35: **Measured online and offline cost.** Online values are mean±SD over 500 edits.

Metric	Vanilla LoRA	SAE-LOCKED LoRA	Overhead
Optimization time/edit	32.4 ± 0.8 s	33.1 ± 0.9 s	2.1%
Feature scoring/edit	0.0 ± 0.0 s	1.5 ± 0.2 s	–
Total wall time/edit	32.4 ± 0.8 s	34.6 ± 1.1 s	6.8%
Peak GPU memory	16.8 ± 0.1 GB	18.2 ± 0.2 GB	8.3%
Offline SAE training	0 GPU-h	120 GPU-h	–

The 2.2-second online increment yields a simple compute-time amortization point of $120 \times 3600/2.2 \approx 196{,}000$ edits. This calculation excludes queueing, storage, and energy costs and is therefore an accounting reference rather than a deployment claim.

N.9 Inference and multiplicity

The analysis unit is the edit instance. Ten random-control draws are first averaged within each optimization seed; the resulting five values define the reported mean and SD. CounterFact bootstrap replicates resample the 2,000 edit clusters and retain those aligned seed-level averages within each selected cluster. Paired effects are computed within edit before aggregation. The family of basis-control and causal-audit contrasts is corrected with the Holm–Bonferroni procedure. Seed-level SD describes optimization variability; edit-clustered intervals describe uncertainty across the evaluated instances and are not interchangeable.

O Safety and Ethics Addendum

This section expands the broader-impact discussion with a dual-use analysis and release recommendations for adaptive feature-constrained model editing. The central concern is that improved control over model updates can support safer maintenance and debugging, while also making targeted behavioral modification easier.

O.1 Dual-use and misuse risks

More controllable edits can reduce unintended behavioral change during legitimate model maintenance, but the same precision may also lower the cost of harmful interventions. Potential misuse includes targeted insertion of false or biased associations, weakening of refusal behavior, and narrow triggered or backdoor-like modifications that may not be detected by coarse aggregate evaluation. Sparse routing may also make some interventions less visible to global drift metrics because the update is concentrated in a small set of representational directions.

These concerns are not unique to SAE-Locked LoRA; they apply broadly to model editing and parameter-efficient adaptation. However, feature-level control increases the importance of evaluating not only whether an edit succeeds, but also what behavior changes outside the intended target. Audit records and locality measurements should therefore be treated as diagnostic evidence rather than guarantees of safety.

O.2 Recommended release practices

We recommend a staged release strategy with defense-in-depth safeguards.

Evaluation-first release. An initial release can prioritize reproducibility by providing code for benchmark evaluation, locality measurement, attribution analysis, and audit-record generation without exposing a turnkey interface for arbitrary target edits. Default configurations should remain restricted to the model families, datasets, and edit settings evaluated in this work.

Safeguarded editing interface. Where an editing interface is provided, requests should be logged together with the selected feature set S_e, attribution scores, adaptive budget K_e, attribution coverage C_e, and relevant in-mask and out-of-mask drift statistics. Deployments involving more capable checkpoints may additionally require access control, rate limits, and review of higher-risk editing requests.

Safety regression testing. Edit success alone is insufficient for deployment. Post-edit evaluation should include locality, broad behavioral drift, refusal behavior, toxicity, bias, jailbreak susceptibility, and any task-specific safety criteria relevant to the deployment setting. A release should proceed only when the edited model satisfies the corresponding safety and behavioral regression requirements.

Artifact release choices. Some artifacts may increase the ease of performing targeted interventions, including SAE dictionaries for highly capable models and automated target-specific scoring or

masking pipelines. Lower-risk release options include aggregate statistics, diagnostic outputs, smaller-model artifacts, or access restricted to research settings.

Operational monitoring. Deployed edits should be monitored for distributional drift, unexpected guardrail failures, and behavioral changes on broad prompt sets. Feature-level audit records indicate where the edit signal was routed and what collateral drift was measured, but they do not establish that all downstream effects have been identified.

O.3 INTENDED USE

The intended use of SAE-LOCKED LORA is controlled model maintenance, debugging, and evaluation, particularly in settings where an update should alter a narrow target behavior while limiting collateral change. The method is not intended as a mechanism for unreviewed arbitrary model modification. Responsible use should combine feature-level diagnostics with conventional safety evaluation, controlled release procedures, and post-edit monitoring.

Minimum provenance record. For reproducible and auditable deployment, each released edit should retain the identifiers or hashes of the model, tokenizer, SAE, adapter, code, and evaluator configuration; the edit instance and random seed; the selected feature set S_e; K_e; attribution scores; coverage and saturation statistics; relevant trainable-path diagnostics; and pre- and post-edit efficacy, locality, KL, uncertainty, and safety measurements. For aggregate metrics such as ReCoE `correct_flip`, the underlying denominators should also be preserved.

Rollback and incident response. Adapters should remain separable from the base model so that an edited state can be disabled without modifying the underlying checkpoint. Deployments should retain the most recent approved artifact and define thresholds for locality, safety, and behavioral drift. If an edit violates these thresholds, the adapter can be disabled, the previous approved state restored, and the associated audit evidence retained for diagnosis. Potential failure sources include unstable feature selection, insufficient mask capacity, uncovered trainable paths, and evaluation drift. A revised edit should be released only after the relevant evaluation suite is rerun successfully.