SAFE DOWNSTREAM ADAPTATION OF LLMS VIA REFUSAL-ORTHOGONAL GATED EDITING

Nayeema Nonta[1,2], **Sirisha Rambhatla**[1,2]
{nnonta, srambhat}@uwaterloo.ca

[1]Critical ML Lab
[2]University of Waterloo

ABSTRACT

Parameter-efficient fine-tuning (PEFT) methods adapt large language models (LLMs) with limited data and compute, but can inadvertently degrade safety-critical refusal behavior. Prior work suggests refusal is concentrated in a small number of directions in representation space; we hypothesize that estimating these directions and restricting adaptation to their orthogonal complement can prevent downstream fine-tuning from making a model less safe. We propose **Safety-Aware Gated Editing (SAGE)**, a two-stage method that (i) estimates a refusal subspace and (ii) fine-tunes sparse, gated additive and multiplicative activation edits for task performance while constraining all edits to the refusal-orthogonal subspace. On Llama-3.1-8B-Instruct, SAGE achieves competitive downstream task and MMLU-Pro accuracy while consistently lowering attack success rates on StrongREJECT, AdvBench, and JailbreakBench compared to standard PEFT and unconstrained activation editing. Overall, SAGE operationalizes refusal directions as a protected control primitive, enabling geometry-aware adaptation that preserves safety under low-resource fine-tuning.

1 INTRODUCTION

Parameter-efficient fine-tuning (PEFT) methods such as LoRA enable large language models (LLMs) to adapt to downstream tasks with limited data and compute. However, even benign task adaptation can unintentionally interfere with safety-critical refusal behavior, increasing susceptibility to jailbreaks or harmful outputs (Qi et al., 2023; Yang et al., 2023; Soligo et al., 2025; Pandey et al., 2025). This raises a central question: how can downstream behavior be adapted in a way that avoids disrupting refusal-sensitive representations?

Existing approaches to safe fine-tuning largely operate in weight or update space, either by incorporating explicit safety objectives during training (Zhang et al., 2024; Zhao et al., 2025a), shaping parameter updates using offline safety information (Lu et al., 2025; Wu et al., 2025), or combining task-tuned and safety-aligned models post-hoc (Djuhera et al., 2025). While effective in some regimes, these methods do not directly address how task adaptation interacts with the internal representations that mediate refusal. Recent work suggests that refusal behavior is concentrated along

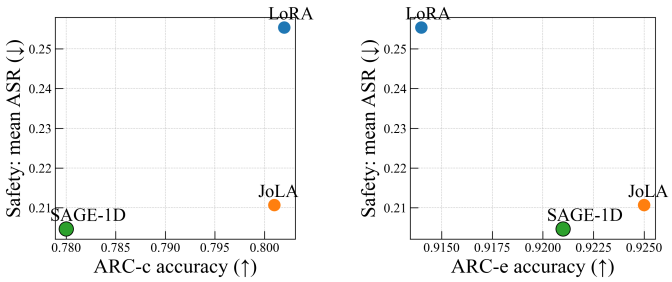


Figure 1: Safety–performance trade-off under low-resource fine-tuning methods on ARC. Safety is the mean attack success rate (ASR) across StrongREJECT, AdvBench, and JailbreakBench.

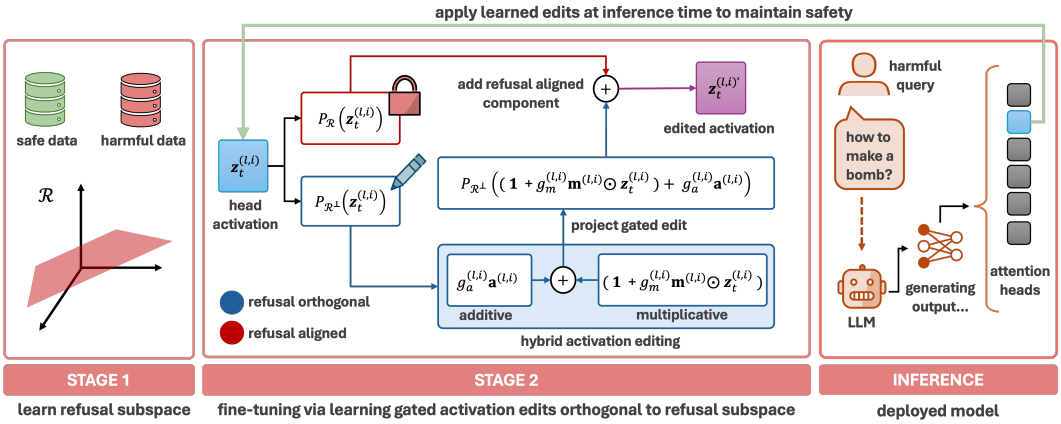


Figure 2: Overview of SAGE. **Stage 1:** Estimate refusal subspace $\mathcal{R}$ from refusal-related behavior. **Stage 2:** Fine-tune for the downstream task using gated activation edits, while constraining edits to lie in $\mathcal{R}^\perp$, with a final projection to prevent reintroducing refusal-aligned directions before recombination. **Inference:** Apply the learned edits at test time when running the fine-tuned model.

structured, low-dimensional directions in representation space (Arditi et al., 2024; Wollschläger et al., 2025; Pan et al., 2025). This motivates a representation-level perspective: safety degradation under fine-tuning may arise when task updates interfere with refusal-sensitive directions.

In this work, we study safe downstream adaptation as a problem of representation-level control. Rather than modifying the task objective or introducing additional safety supervision, we ask whether downstream adaptation can be constrained to avoid modifying refusal-related representations. Concretely, we treat refusal as a behaviorally grounded representation that can be estimated and selectively protected during fine-tuning.

At each layer l, we index editable components (e.g., attention heads) by i and denote their activation at token position t as $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$. Following prior evidence (Arditi et al., 2024; Wollschläger et al., 2025), we model refusal as a low-dimensional linear subspace $\mathcal{R} \subset \mathbb{R}^d$. This induces the decomposition

$$\mathbf{z}_t^{(l,i)} = P_{\mathcal{R}}\, \mathbf{z}_t^{(l,i)} + P_{\mathcal{R}^\perp}\, \mathbf{z}_t^{(l,i)}, \tag{1}$$

where $P_{\mathcal{R}}$ and $P_{\mathcal{R}^\perp}$ denote projection onto the refusal subspace and its orthogonal complement, respectively. Intuitively, $P_{\mathcal{R}}\, \mathbf{z}_t^{(l,i)}$ captures refusal-relevant information that should remain stable, while $P_{\mathcal{R}^\perp}\, \mathbf{z}_t^{(l,i)}$ represents degrees of freedom available for downstream adaptation.

To parameterize controlled adaptation, we adopt gated additive and multiplicative activation edits following JoLA (Lai et al., 2025). For each editable component (l, i), we associate additive and multiplicative edit vectors $\mathbf{a}^{(l,i)}, \mathbf{m}^{(l,i)} \in \mathbb{R}^d$, together with scalar gates $g_a^{(l,i)}, g_m^{(l,i)} \in [0, 1]$ that encourage sparse, component-wise editing. Gates are learned jointly with the edit parameters and replaced by their expected values at inference time.

We propose **Safety-Aware Gated Editing (SAGE)**, a two-stage method that leverages the decomposition in Eq. 1 to enable safe downstream adaptation. In Stage 1, SAGE estimates a basis for the refusal subspace $\mathcal{R}$. In Stage 2, we perform gated activation editing for task adaptation while constraining all learned edits to the refusal-orthogonal subspace $\mathcal{R}^\perp$. Edit parameters are projected onto $\mathcal{R}^\perp$, and post-edit activations are re-projected to prevent the reintroduction of refusal-aligned components. As a result, refusal-related activations are preserved, while downstream adaptation occurs exclusively in orthogonal directions.

2 Related Work

Activation editing for parameter-efficient adaptation. Recent work explores parameter-efficient adaptation via activation- or representation-space edits rather than full weight updates.

JoLA (Lai et al., 2025) introduces sparse, gated additive and multiplicative activation edits optimized under a task loss with an L_0 sparsity penalty. Related approaches include bias-only or scaling-based edits (e.g., BitFit (Ben Zaken et al., 2022), RED (Wu et al., 2024a)), low-rank or head-selective interventions (LoFIT (Yin et al., 2024)), and subspace-restricted activation editing (ReFT (Wu et al., 2024b)). These methods focus on efficiency and localization, but do not explicitly address safety preservation during downstream adaptation. SAGE builds on gated activation editing and introduces a geometric constraint to control how adaptation interacts with safety-relevant representations.

Refusal geometry and representation-level safety. Early work identifies single refusal directions (Arditi et al., 2024), while subsequent work uncovers higher-dimensional refusal spaces and proposes Refusal Direction Optimization (RDO) to learn layer-wise refusal vectors (Wollschläger et al., 2025; Pan et al., 2025). Many prior methods (Nguyen et al., 2025; Li et al., 2025; Oozeer et al., 2025) use these directions for steering, ablation, or inference-time intervention. In contrast, SAGE uses refusal geometry as a constraint on where downstream adaptation is allowed to occur, treating refusal-aligned directions as protected during fine-tuning.

Safe fine-tuning in weight space. Weight-space approaches preserve safety by modifying objectives, updates, or post-training model composition. These include explicit safety-regularized or preference-based objectives (Zhang et al., 2024; Zhao et al., 2025a), curvature- or probe-informed update selection (Lu et al., 2025; Wu et al., 2025), and post-hoc layer merging between task-tuned and aligned models (Djuhera et al., 2025). Unlike these methods, SAGE introduces no explicit safety loss and does not constrain weight updates directly. Instead, it enforces safety preservation through a lightweight, representation-level geometric constraint during representation-based adaptation.

3 Background

Activations and refusal subspace. Let $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$ denote the activation of component (l, i) at token position t. Following prior work, we model refusal behavior as being mediated by a low-dimensional subspace $\mathcal{R} \subset \mathbb{R}^d$. This induces the orthogonal decomposition in Eq. 1, where $P_{\mathcal{R}}$ and $P_{\mathcal{R}^\perp}$ denote projection onto the refusal subspace and its orthogonal complement.

Gated activation editing. We adopt gated additive and multiplicative activation edits following JoLA (Lai et al., 2025). For each editable component (l, i), we associate additive and multiplicative edit vectors $\mathbf{a}^{(l,i)}, \mathbf{m}^{(l,i)} \in \mathbb{R}^d$ with scalar gates $g_a^{(l,i)}, g_m^{(l,i)} \in [0, 1]$. The gated hybrid edit is

$$\mathbf{z}_t^{(l,i)\prime} = \left(\mathbf{1} + g_m^{(l,i)}\mathbf{m}^{(l,i)}\right) \odot \mathbf{z}_t^{(l,i)} + g_a^{(l,i)}\mathbf{a}^{(l,i)}, \tag{2}$$

where the gates encourage sparse, localized adaptation. Full details are provided in Appendix B.

Estimating refusal geometry. Refusal directions are estimated using prior methods such as difference-in-means or Refusal Direction Optimization (RDO) (Wollschläger et al., 2025), which identify directions along which additive or ablative interventions reliably toggle refusal behavior. We estimate a low-dimensional refusal subspace spanned by a small set of such directions. Full details of refusal geometry estimation are provided in Appendix B.4.

4 Method

We propose SAGE , a two-stage activation-editing procedure for safe downstream adaptation. First, we estimate a refusal subspace on a frozen base model. Second, we fine-tune sparse, gated activation edits for a downstream task while constraining all edits to lie in the orthogonal complement of this subspace. Algorithm 1 summarizes the procedure. See Appendices B and C for details.

Stage 1: Estimating the refusal subspace. Let f_θ be a frozen base model. Following prior work on refusal geometry (Wollschläger et al., 2025), we estimate a low-dimensional refusal basis $B \in \mathbb{R}^{d\times N}$ using Refusal Cone Optimization (RCO). We define the induced refusal subspace and associated projectors as

$$\mathcal{R} := \operatorname{span}(B), \qquad P_{\mathcal{R}} := BB^\top, \qquad P_{\mathcal{R}^\perp} := I - BB^\top. \tag{3}$$

Stage 2: Refusal-orthogonal gated editing. Let $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$ denote the activation of component (l, i) at token position t. With the refusal subspace fixed, we decompose each activation as described in Eq.1. We apply gated additive and multiplicative edits only to the refusal-orthogonal component. Before each forward pass, unconstrained edit parameters are projected into $\mathcal{R}^\perp$, and the post-edit activation is re-projected to prevent reintroducing refusal-aligned components. The resulting edited activation is

$$\mathbf{z}_t^{(l,i)\prime} = P_{\mathcal{R}}\mathbf{z}_t^{(l,i)} + P_{\mathcal{R}^\perp}\Big((\mathbf{1} + g_m^{(l,i)}\mathbf{m}^{(l,i)}) \odot P_{\mathcal{R}^\perp}\mathbf{z}_t^{(l,i)} + g_a^{(l,i)}\mathbf{a}^{(l,i)}\Big). \tag{4}$$

Thus, refusal-related activations are preserved exactly, while downstream adaptation occurs only in orthogonal directions.

Training objective. We optimize only the edit and gate parameters using the standard task cross-entropy loss together with an L_0 sparsity penalty on the gates (Lai et al., 2025):

$$\min_{\phi} \; \mathcal{L}_{\text{CE}}(\theta, \phi; \mathcal{D}) \; + \; \rho\, \Omega_{L_0}(\psi), \tag{5}$$

keeping the base model parameters θ and refusal subspace fixed.

Algorithm 1 SAGE: Safety-Aware Gated Editing

Require: Base model f_θ; refusal datasets $\mathcal{D}_{\text{harm}}, \mathcal{D}_{\text{safe}}$ (Stage 1); task data $\mathcal{D}_{\text{ben}}$ (Stage 2); refusal subspace dimension N; samples per iteration M; sparsity weight ρ

Ensure: Refusal basis $B \in \mathbb{R}^{d\times N}$ and projectors $P_{\mathcal{R}}, P_{\mathcal{R}^\perp}$; edit parameters $\phi = \{\tilde{\mathbf{m}}^{(l,i)}, \tilde{\mathbf{a}}^{(l,i)}\}$ and gate parameters $\psi = \{\psi_m^{(l,i)}, \psi_a^{(l,i)}\}$.

Notation: $\mathcal{R} = \text{span}(B)$; $P_{\mathcal{R}} = BB^\top$, $P_{\mathcal{R}^\perp} = I - BB^\top$. $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$ denotes the activation of component (l, i) at token position t.

// STAGE 1: LEARN REFUSAL BASIS

Initialize $B \in \mathbb{R}^{d\times N}$ with $B^\top B = I$
for iter $= 1 \dots K$ **do**
  Sample $\{\boldsymbol{\lambda}^{(m)}\}_{m=1}^M$ with $\boldsymbol{\lambda}^{(m)} \in \mathbb{R}^N_{\geq 0}$ {M candidate directions sampled from the refusal cone.}
  for $m = 1 \dots M$ **do**
    $\mathbf{r}_m \leftarrow B\boldsymbol{\lambda}^{(m)}$; $\hat{\mathbf{r}}_m \leftarrow \mathbf{r}_m / \|\mathbf{r}_m\|_2$
    $\mathcal{L}_m \leftarrow \mathcal{L}_{\text{RDO}}(\hat{\mathbf{r}}_m; \mathcal{D}_{\text{harm}}, \mathcal{D}_{\text{safe}})$ {Refusal direction optimization (RDO) (Appendix B.4, Eq. 17).}
  end for
  $\mathcal{L}_{\text{RCO}} \leftarrow \frac{1}{M}\sum_{m=1}^M \mathcal{L}_m$ {Refusal cone optimization (RCO) (Appendix B.4, Eq. 23).}
  $B \leftarrow B - \eta \nabla_B \mathcal{L}_{\text{RCO}}$
  Orthonormalize columns of B
end for
Freeze B; define the refusal subspace $\mathcal{R} := \text{span}(B)$; define projectors $P_{\mathcal{R}} = BB^\top$, $P_{\mathcal{R}^\perp} = I - BB^\top$

// STAGE 2: REFUSAL-ORTHOGONAL GATED ADAPTATION

Choose editable components $C = \{(l, i)\}$ {e.g., selected attention heads.}
for each $(l, i) \in C$ **do**
  Initialize $\tilde{\mathbf{m}}^{(l,i)}, \tilde{\mathbf{a}}^{(l,i)} \in \mathbb{R}^d$, $\psi_m^{(l,i)}, \psi_a^{(l,i)}$
end for
for iter $= 1 \dots K'$ **do**
  Sample $(x, y) \sim \mathcal{D}_{\text{ben}}$
  Forward f_θ on x
  for each $(l, i) \in C$ **do**
    $\mathbf{m}^{(l,i)} \leftarrow P_{\mathcal{R}^\perp}\tilde{\mathbf{m}}^{(l,i)}$; $\mathbf{a}^{(l,i)} \leftarrow P_{\mathcal{R}^\perp}\tilde{\mathbf{a}}^{(l,i)}$
    $g_m^{(l,i)}, g_a^{(l,i)} \leftarrow \text{HC}(\psi_m^{(l,i)}, \psi_a^{(l,i)})$ {Hard–Concrete gates in [0, 1]; mean used at inference.}
    $\mathbf{z}_{\mathcal{R}} \leftarrow P_{\mathcal{R}}(\mathbf{z}_t^{(l,i)})$; $\mathbf{z}_\perp \leftarrow P_{\mathcal{R}^\perp}(\mathbf{z}_t^{(l,i)})$
    $\mathbf{z}_t^{(l,i)\prime} \leftarrow \mathbf{z}_{\mathcal{R}} + P_{\mathcal{R}^\perp}\Big((\mathbf{1} + g_m^{(l,i)}\mathbf{m}^{(l,i)}) \odot \mathbf{z}_\perp + g_a^{(l,i)}\mathbf{a}^{(l,i)}\Big)$
  end for
  $\mathcal{L} \leftarrow \text{CE}(f_{\theta,\phi}(x; B), y) + \rho\,\Omega_{L_0}(\psi)$
  Freeze θ, B; update (ϕ, ψ) with learning rate η'
end for

Table 1: Results on Llama-3.1-8B-Instruct. We report accuracy (↑) on ARC-c/ARC-e and capability retention on MMLU-Pro, and safety robustness (↓) via StrongREJECT score (SR) and attack success rate (ASR) on AdvBench and JailbreakBench. All fine-tuned runs use 200 training examples.

FT task	Method	ARC-c↑	ARC-e↑	MMLU-Pro↑	SR↓	AdvBench↓	JailbreakBench↓
None	zero-shot	0.765	0.883	0.377	0.220	0.231	0.320
ARC-c	LoRA	0.802	0.914	0.341	0.207	0.229	0.330
ARC-c	JoLA	0.801	0.925	0.302	0.156	0.225	0.251
ARC-c	SAGE-1D (Ours)	0.780	0.921	0.343	0.151	0.218	0.245
ARC-c	SAGE-5D (Ours)	0.780	0.920	0.345	0.152	0.219	0.251

5 Experimental Setup

We study Llama-3.1-8B-Instruct; each run uses a single NVIDIA A100-SXM4-80GB GPU. We estimate the refusal basis with the RCO procedure and hyperparameters of Wollschläger et al. (2025), using a mix of harmful and harmless prompts from Alpaca (Taori et al., 2023) and SALAD-BENCH (Li et al., 2024). Unless noted, we fine-tune on ARC-Challenge (ARC-c) (Clark et al., 2018) with 200 training examples and compare zero-shot, LoRA, JoLA (Lai et al., 2025), and SAGE (ours) (Table 1); full training settings are in Appendix D. We report accuracy on ARC-c/ARC-e (Clark et al., 2018) and capability retention on MMLU-Pro (Wang et al., 2024) (no training on MMLU-Pro). Safety is evaluated on StrongREJECT (Souly et al., 2024), AdvBench (Zou et al., 2023b), and JailbreakBench (Chao et al., 2024); ARC/MMLU-Pro answers are extracted via deterministic regex, and safety uses StrongREJECT score plus ASR (lower is better) on AdvBench/JailbreakBench, with the StrongREJECT evaluator as the judge across all safety benchmarks.

6 Results

All adaptation methods improve ARC-c accuracy over zero-shot (Table 1). LoRA and JoLA achieve the largest in-domain gains, while SAGE-1D incurs a modest reduction relative to JoLA on ARC-c (0.780 vs. 0.801) while remaining competitive on ARC-e. Figure 1 summarizes the safety–performance trade-off: SAGE attains lower mean attack success rates than JoLA across all safety benchmarks, whereas LoRA exhibits a clearer trade-off between task accuracy and robustness.

Safety metrics further show that benign fine-tuning can alter harmfulness in non-monotonic ways. While some fine-tuned models improve on individual safety benchmarks relative to zero-shot, they may remain vulnerable to jailbreak and refusal-bypass attacks. Consistent with prior observations (Wu et al., 2025; Shah et al., 2025), we find that standard LoRA slightly improves some safety scores but remains susceptible to JailbreakBench attacks.

7 Discussion and Conclusion

SAGE frames safe downstream adaptation as a representation-level control problem: by estimating a refusal subspace on a frozen base model and constraining all gated activation edits to its orthogonal complement, refusal-sensitive components are preserved during fine-tuning. In a low-resource regime on Llama-3.1-8B-Instruct (200 examples), this geometric constraint enables SAGE to retain competitive downstream accuracy and general capability while consistently reducing attack success rates relative to unconstrained activation editing. The effect of the refusal constraint on gate activity is reported in Appendix E, indicating that these gains are not explained by suppressing adaptation.

A key finding is that increasing the protected refusal subspace from 1D to 5D does not improve safety, plausibly because multi-direction constraints incorporate weaker or noisier axes that dilute the effective constraint and increase variance (Wollschläger et al., 2025). Overall, SAGE operationalizes refusal geometry as a protected control primitive, demonstrating that lightweight, representation-level constraints can improve robustness under low-resource fine-tuning. Future work should evaluate broader tasks, domains, and model scales, and examine how improved refusal-subspace estimation impacts robustness.

REFERENCES

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083, 2024.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.1. URL https://aclanthology.org/2022.acl-short.1/.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. JailbreakBench: An open robustness benchmark for jailbreaking large language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=urjPCYZt0I.

Jianhui Chen, Xiaozhi Wang, Zijun Yao, Yushi Bai, Lei Hou, and Juanzi Li. Finding safety neurons in large language models. *arXiv preprint arXiv:2406.14144*, 2024.

Hyeong Kyu Choi, Xuefeng Du, and Yixuan Li. Safety-aware fine-tuning of large language models. *arXiv preprint arXiv:2410.10014*, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Aladin Djuhera, Swanand Ravindra Kadhe, Farhan Ahmed, Syed Zawad, and Holger Boche. Safemerge: Preserving safety alignment in fine-tuned large language models via selective layer-wise model merging. *arXiv preprint arXiv:2503.17239*, 2025.

Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. Safe lora: The silver lining of reducing safety risks when finetuning large language models. *Advances in Neural Information Processing Systems*, 37:65072–65094, 2024.

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Tekin, and Ling Liu. Lisa: Lazy safety alignment for large language models against harmful fine-tuning attack. *Advances in Neural Information Processing Systems*, 37:104521–104555, 2024.

Wen Lai, Alexander Fraser, and Ivan Titov. Joint localization and activation editing for low-resource fine-tuning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Lllg9YjAFX.

Lijun Li, Bowen Dong, Ruohui Wang, Xuhao Hu, Wangmeng Zuo, Dahua Lin, Yu Qiao, and Jing Shao. Salad-bench: A hierarchical and comprehensive safety benchmark for large language models. *arXiv preprint arXiv:2402.05044*, 2024.

Yichen Li, Zhiting Fan, Ruizhe Chen, Xiaotang Gai, Luqi Gong, Yan Zhang, and Zuozhu Liu. Fairsteer: Inference time debiasing for llms with dynamic activation steering. *arXiv preprint arXiv:2504.14492*, 2025.

Xiaoqun Liu, Jiacheng Liang, Muchao Ye, and Zhaohan Xi. Robustifying safety-aligned large language models through clean data curation. *arXiv preprint arXiv:2405.19358*, 2024.

Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through l_0 regularization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1Y8hhg0b.

Ning Lu, Shengcai Liu, Jiahao Wu, Weiyu Chen, Zhirui Zhang, Yew-Soon Ong, Qi Wang, and Ke Tang. Safe delta: Consistently preserving safety when fine-tuning llms on diverse datasets. *arXiv preprint arXiv:2505.12038*, 2025.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*, 2023.

Duy Nguyen, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. Multi-attribute steering of language models via targeted intervention. *arXiv preprint arXiv:2502.12446*, 2025.

Narmeen Oozeer, Luke Marks, Fazl Barez, and Amirali Abdullah. Beyond linear steering: Unified multi-attribute control for language models. *arXiv preprint arXiv:2505.24535*, 2025.

Wenbo Pan, Zhichao Liu, Qiguang Chen, Xiangyang Zhou, Haining Yu, and Xiaohua Jia. The hidden dimensions of llm alignment: A multi-dimensional analysis of orthogonal safety directions. *arXiv preprint arXiv:2502.09674*, 2025.

Punya Syon Pandey, Samuel Simko, Kellin Pelrine, and Zhijing Jin. Accidental vulnerability: Factors in fine-tuning that shift model safeguards. In *Workshop on Socially Responsible Language Modelling Research*, 2025.

ShengYun Peng, Pin-Yu Chen, Jianfeng Chi, Seongmin Lee, and Duen Horng Chau. Shape it up! restoring llm safety during finetuning. *arXiv preprint arXiv:2505.17196*, 2025.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*, 2023.

Shaan Shah, Kaustubh Ponkshe, Raghav Singhal, and Praneeth Vepakomma. Safety subspaces are not distinct: A fine-tuning case study. In *The First Workshop on the Interplay of Model Behavior and Model Internals*, 2025. URL https://openreview.net/forum?id=2uLBkfMyX5.

Anna Soligo, Edward Turner, Senthooran Rajamanoharan, and Neel Nanda. Convergent linear representations of emergent misalignment. *URL https://arxiv. org/abs/2506.11618*, 2025.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. A strongREJECT for empty jailbreaks. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=KZLE5BaaOH.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*, 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

Tom Wollschläger, Jannes Elstner, Simon Geisler, Vincent Cohen-Addad, Stephan Günnemann, and Johannes Gasteiger. The geometry of refusal in large language models: Concept cones and representational independence. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=80IwJqlXs8.

Chengcan Wu, Zhixin Zhang, Zeming Wei, Yihao Zhang, and Meng Sun. Mitigating fine-tuning risks in llms via safety-aware probing optimization. *arXiv preprint arXiv:2505.16737*, 2025.

Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Zhu JianHao, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. Advancing parameter efficiency in fine-tuning via representation editing. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13445–13464, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.726. URL https://aclanthology.org/2024.acl-long.726/.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. Reft: Representation finetuning for 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. 63908–63962. Curran Associates, Inc., 2024b. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/75008a0fba53bf13b0bb3b7bff986e0e-Paper-Conference.pdf.

Xianjun Yang, Xiao Wang, Qi Zhang, Linda Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. Shadow alignment: The ease of subverting safely-aligned language models. *arXiv preprint arXiv:2310.02949*, 2023.

Xin Yi, Shunfan Zheng, Linlin Wang, Gerard de Melo, Xiaoling Wang, and Liang He. Nlsr: Neuron-level safety realignment of large language models against harmful fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 25706–25714, 2025.

Fangcong Yin, Xi Ye, and Greg Durrett. Lofit: Localized fine-tuning on llm representations. 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. 9474–9506. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/122ea6470232ee5e79a2649243348005-Paper-Conference.pdf.

Lei Yu, Virginie Do, Karen Hambardzumyan, and Nicola Cancedda. Robust llm safeguarding via refusal feature adversarial training. *arXiv preprint arXiv:2409.20089*, 2024.

Wenxuan Zhang, Philip HS Torr, Mohamed Elhoseiny, and Adel Bibi. Bi-factorial preference optimization: Balancing safety-helpfulness in language models. *arXiv preprint arXiv:2408.15313*, 2024.

Weixiang Zhao, Yulin Hu, Yang Deng, Jiahe Guo, Xingyu Sui, Xinyang Han, An Zhang, Yanyan Zhao, Bing Qin, Tat-Seng Chua, et al. Beware of your po! measuring and mitigating ai safety risks in role-play fine-tuning of llms. *arXiv preprint arXiv:2502.20968*, 2025a.

Yiran Zhao, Wenxuan Zhang, Yuxi Xie, Anirudh Goyal, Kenji Kawaguchi, and Michael Shieh. Understanding and enhancing safety mechanisms of llms via safety-specific neuron. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023b. URL https://arxiv.org/abs/2307.15043.

A Related Work

Scope and taxonomy. Methods for safe fine-tuning differ along three key axes: (i) *where* the intervention is applied (weight/parameter space vs. activation/representation space), (ii) *when* safety information is incorporated (e.g., offline parameter selection, during fine-tuning, or post-hoc), and (iii) *how* safety signals influence adaptation (e.g., explicit loss term, parameter selection, post-hoc layer merging, or geometric constraints). Methods also vary in the degree of localization or sparsity of the intervention, ranging from full-model updates to sparse, component-level modifications, as well as the mechanism used to induce sparsity (e.g., low-rank parameterization, gating, or layer/head selection). Our method, SAGE, is a training-time, activation-space approach that applies sparse, localized edits gated by Hard–Concrete distributions and constrains these edits using a refusal-orthogonality constraint.

Activation editing for parameter-efficient fine-tuning. Lai et al. (2025) propose joint localization and activation editing (JoLA): lightweight, training-time edits applied to selected components (e.g., attention heads) with Hard–Concrete gating to induce sparsity, optimized under task cross-entropy plus an expected L_0 penalty. Adjacent training-time activation-editing approaches include BitFit (bias-only tuning) (Ben Zaken et al., 2022), RED (layer-wise scaling and bias vectors) (Wu et al., 2024a), ReFT direct hidden-state intervention that edits only a learned low-rank subspace via an orthonormal projector (Wu et al., 2024b), and LoFIT (head selection + bias tuning) (Yin et al., 2024). Inference-time activation modification has also been explored, e.g., RePE (Zou et al., 2023a). Relative to these works, our proposed method retains JoLA's low-resource, gated edit parameterization but introduces a geometric safety constraint.

Refusal geometry in the representation space. Several works suggest that refusal behavior can be captured by low-dimensional directions in representation space. Arditi et al. (2024) argue for a single refusal direction, while Wollschläger et al. (2025) discover refusal cones and introduce Refusal Direction Optimization (RDO) to learn layer-specific refusal vectors using CE losses for additive/ablation interventions and a KL retain term. Pan et al. (2025) further study multi-dimensional (orthogonal) safety subspaces. Our method directly builds on this line of work: we estimate layer-wise refusal directions using RDO and use the best one to constrain downstream activation edits, rather than steering or patching representations directly.

Activation-space approaches to safety interventions. A growing line of work manipulates internal activations to improve safety, either by patching specific neurons/circuits or by steering along control directions. Neuron-level safety realignment and identification of "safety neurons" have been studied in Yi et al. (2025); Zhao et al. (2025b); Chen et al. (2024). Activation steering methods control attributes by adding learned vectors during inference or training, including multi-attribute steering and dynamic (context-dependent) debiasing (Nguyen et al., 2025; Li et al., 2025; Oozeer et al., 2025). Our method differs in that it learns a refusal geometry (via RDO) and then trains sparse edits that are orthogonal to that geometry at the edit site, aiming to preserve refusal while improving benign utility.

Weight-space approaches to safe fine-tuning. These methods preserve safety by modifying or selecting parameter updates, through explicit safety losses during training, gradient-based update steering using safety probes, or post-hoc layer merging.

The first class of methods incorporate safety information directly in the fine-tuning objective by adding explicit safety terms or preference-based supervision. For example, SaRFT introduces a safety regularizer (e.g., KL-based) during role-play fine-tuning to reduce unsafe behaviors (Zhao et al., 2025a), and joint safety–helpfulness preference optimization optimizes a preference objective that explicitly trades off helpfulness and safety (Zhang et al., 2024). Dynamic Safety Shaping (DSS) further refines this approach by using token-level guardrail feedback to modulate the loss within a single response, reinforcing safe segments while suppressing unsafe ones during training (Peng et al., 2025). Other methods also incorporate safety/alignment signals during training to improve robustness to harmful fine-tuning, such as LISA (Huang et al., 2024) and refusal-feature adversarial training (Yu et al., 2024).

The second class uses safety supervision to determine which updates are safe to apply, without augmenting the task loss with an explicit safety penalty. SafeDelta uses a safety dataset to estimate parameter sensitivity to a safety objective via a second-order (Hessian-informed) approximation, then selects and compensates parameter deltas to bound safety loss change under downstream fine-tuning (Lu et al., 2025). In contrast, safety-aware probing methods estimate harmful *gradient* directions using safety supervision and use these probes to steer or constrain the fine-tuning trajectory in a bi-level or constrained optimization procedure (Wu et al., 2025). SafeLoRA provides a related weight-space constraint by projecting LoRA updates toward a safety-aligned subspace computed from aligned and unaligned model weights (Hsu et al., 2024).

The third class preserves safety via post-hoc operations on weights. SafeMerge fine-tunes a task model and a safety-aligned model separately, then selectively merges layers guided by projection-based similarity to retain safety-critical parameters (Djuhera et al., 2025). In this setting, safety information is used after training to decide which layers or parameters to keep from the aligned reference.

A related class acts before optimization by curating or filtering the fine-tuning data to prevent unsafe gradients from being induced in the first place. Data curation/filtering approaches (e.g., SAFT, CTRL) aim to reduce harmful gradients by selecting safe training instances or removing unsafe segments from the fine-tuning distribution (Choi et al., 2024; Liu et al., 2024).These methods incorporate safety information at the data level rather than through update constraints or modified objectives.

B BACKGROUND

B.1 NOTATION

Let $\mathbf{s} = (s_1, \ldots, s_T)$ be an input token sequence of length T over a vocabulary. We consider a decoder-only Transformer with L layers and hidden size d. For each layer $l \in \{1, \ldots, L\}$ and token position $t \in \{1, \ldots, T\}$ we denote by $\mathbf{x}_t^{(l)} \in \mathbb{R}^d$ the residual stream entering layer l at position t. Each layer consists of an attention block followed by an MLP block, both with residual connections:

$$\tilde{\mathbf{x}}_t^{(l)} = \mathbf{x}_t^{(l)} + \mathrm{ATTN}^{(l)}\left(\mathbf{x}_{1:t}^{(l)}\right), \tag{6}$$

$$\mathbf{x}_t^{(l+1)} = \tilde{\mathbf{x}}_t^{(l)} + \mathrm{MLP}^{(l)}\left(\tilde{\mathbf{x}}_t^{(l)}\right). \tag{7}$$

As in prior work, we omit layer normalization and other architectural details.

At each layer l we index individual components (e.g. attention heads, MLP blocks) by i. We write the activation of component (l, i) at position t as

$$\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d.$$

These $\mathbf{z}_t^{(l,i)}$ are the intermediate activations we will edit.

We write $\|\cdot\|_2$ for the Euclidean norm. A refusal direction is a vector $\mathbf{r} \in \mathbb{R}^d$ with unit vector $\hat{\mathbf{r}} = \mathbf{r}/\|\mathbf{r}\|_2$. This gives the projection operators $P_{\hat{\mathbf{r}}} = \hat{\mathbf{r}}\hat{\mathbf{r}}^\top$ and $P_{\hat{\mathbf{r}}^\perp} = I - \hat{\mathbf{r}}\hat{\mathbf{r}}^\top$. Any component activation then decomposes as

$$\mathbf{z}_t^{(l,i)} = P_{\hat{\mathbf{r}}}\, \mathbf{z}_t^{(l,i)} + P_{\hat{\mathbf{r}}^\perp}\, \mathbf{z}_t^{(l,i)}.$$

We write f_θ for the base model with parameters θ. When we introduce additional lightweight edit parameters we write $f_{\theta,\phi}$, keeping ϕ (edit parameters) separate from the refusal geometry (e.g. $\hat{\mathbf{r}}$ or a basis B).

B.2 INTERVENTION TYPES

Activation intervention types. Activation editing modifies intermediate activations $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$ for component (l, i) at token position t. Following Lai et al. (2025), we consider three classical edit types. Let $\mathbf{a}^{(l,i)}, \mathbf{m}^{(l,i)} \in \mathbb{R}^d$ denote additive and multiplicative edit parameters, and let $\odot$ denote element-wise multiplication.

- **Additive:** add a bias vector
$$\mathbf{z}_t^{(l,i)\prime} = \mathbf{z}_t^{(l,i)} + \mathbf{a}^{(l,i)}.$$
- **Multiplicative:** element-wise rescaling
$$\mathbf{z}_t^{(l,i)\prime} = \mathbf{m}^{(l,i)} \odot \mathbf{z}_t^{(l,i)}.$$
- **Hybrid:** combine both transformations
$$\mathbf{z}_t^{(l,i)\prime} = \mathbf{m}^{(l,i)} \odot \mathbf{z}_t^{(l,i)} + \mathbf{a}^{(l,i)}.$$

Refusal intervention in activation space. Let $\hat{\mathbf{r}} := \mathbf{r}/\|\mathbf{r}\|_2$ denote a unit refusal direction. Following Wollschläger et al. (2025), we consider two interventions applied to residual activations $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$ (at selected layers/components and token positions):

$$\text{Add:} \quad \mathbf{z}_t^{(l,i)\prime} = \mathbf{z}_t^{(l,i)} + \alpha\, \hat{\mathbf{r}}, \tag{8}$$

$$\text{Ablate:} \quad \mathbf{z}_t^{(l,i)\prime} = P_{\hat{\mathbf{r}}^\perp}\, \mathbf{z}_t^{(l,i)} = (I - \hat{\mathbf{r}}\hat{\mathbf{r}}^\top)\, \mathbf{z}_t^{(l,i)}. \tag{9}$$

Choosing $\alpha < 0$ in equation 8 corresponds to subtraction along $\hat{\mathbf{r}}$.

B.3 Gated Activation Editing

Lai et al. (2025) extend the hybrid edit by introducing stochastic gates $g_a^{(l,i)}, g_m^{(l,i)} \in [0,1]$, which independently enable additive and multiplicative edits. Let $\mathbf{a}^{(l,i)}, \mathbf{m}^{(l,i)} \in \mathbb{R}^d$ denote additive and multiplicative edit parameters. The gated hybrid edit is

$$\mathbf{z}_t^{(l,i)\prime} = \big(\mathbf{1} + g_m^{(l,i)}\mathbf{m}^{(l,i)}\big) \odot \mathbf{z}_t^{(l,i)} + g_a^{(l,i)}\mathbf{a}^{(l,i)}, \tag{10}$$

where $\mathbf{1} \in \mathbb{R}^d$ is a vector of ones. When both gates are zero ($g_a^{(l,i)} = g_m^{(l,i)} = 0$), the edit reduces to the identity, i.e., $\mathbf{z}_t^{(l,i)\prime} = \mathbf{z}_t^{(l,i)}$. Each gate is drawn from a Hard–Concrete distribution (Louizos et al., 2018), a mixed discrete–continuous distribution on $[0,1]$ with point masses at $\{0,1\}$ and a continuous density on $(0,1)$. At inference time, we replace each stochastic gate by its mean, setting $g_a^{(l,i)} = \mathbb{E}[g_a^{(l,i)}]$ and $g_m^{(l,i)} = \mathbb{E}[g_m^{(l,i)}]$ to ensure determinism.

Hard–Concrete sparsity penalty. Following Lai et al. (2025), each gate $g_a^{(l,i)}, g_m^{(l,i)}$ is parameterized by Hard–Concrete parameters $\psi_a^{(l,i)}, \psi_m^{(l,i)}$. The expected L_0 penalty is

$$\Omega_{L_0}(\psi) = \sum_{l,i} \Big(\Pr(g_a^{(l,i)} \neq 0 \mid \psi_a^{(l,i)}) + \Pr(g_m^{(l,i)} \neq 0 \mid \psi_m^{(l,i)}) \Big). \tag{11}$$

B.4 Refusal Geometry Discovery

Difference-in-Means (DIM). The difference-in-means (DIM) method (Arditi et al., 2024; Marks & Tegmark, 2023; Tigges et al., 2023) estimates a linear refusal direction from mean activations of harmful and harmless prompts.

Let D_{harm} and D_{safe} be datasets of harmful and harmless prompts, respectively. For a prompt p, let $\mathbf{z}_t^{(l,i)}(p) \in \mathbb{R}^d$ denote the activation at layer l, component (l,i), and token position t when running f_θ on p, where t is a fixed extraction position used consistently across prompts. The class-conditional mean activations are

$$\begin{aligned} \boldsymbol{\mu}^{(l,i)} &= \frac{1}{|D_{\text{harm}}|} \sum_{p \in D_{\text{harm}}} \mathbf{z}_t^{(l,i)}(p), \\ \boldsymbol{\nu}^{(l,i)} &= \frac{1}{|D_{\text{safe}}|} \sum_{p \in D_{\text{safe}}} \mathbf{z}_t^{(l,i)}(p). \end{aligned} \tag{12}$$

The resulting difference-in-means vector is

$$\mathbf{r}^{(l,i)} = \boldsymbol{\mu}^{(l,i)} - \boldsymbol{\nu}^{(l,i)}. \tag{13}$$

For interventions we use the unit direction.

Refusal Direction Optimization (RDO). DIM finds a direction that separates harmful from harmless activations, but it does not guarantee that editing along that direction actually produces the behaviors we want. Refusal Direction Optimization (RDO) addresses this by directly optimizing, via a gradient-based loss, a unit vector $\hat{\mathbf{r}} \in \mathbb{R}^d$ so that the two edits we defined in §B.2 (additive steering and directional ablation) reliably toggle refusal.

Let D_{harm} and D_{safe} be datasets of harmful and benign prompts, respectively. RDO defines two modified forward passes of the base model f_θ: $f_{\text{add}}(p; \hat{\mathbf{r}})$, which runs f_θ but adds $\alpha \cdot \hat{\mathbf{r}}$ at a chosen layer l, and $f_{\text{abl}}(p; \hat{\mathbf{r}})$, which runs f_θ but ablates the activation along $\hat{\mathbf{r}}$ using $I - \hat{\mathbf{r}}\hat{\mathbf{r}}^\top$. RDO trains the add pass on benign prompts toward a refusal target, trains the ablate pass on harmful prompts toward an answering target, and adds a retain KL on benign prompts to keep outputs close to f_θ.

$$\mathcal{L}_{\text{add}}(\hat{\mathbf{r}}) = \frac{1}{|D_{\text{safe}}|} \sum_{p \in D_{\text{safe}}} \text{CE}\big(f_{\text{add}}(p; \hat{\mathbf{r}}),\ y_{\text{refuse}}\big), \tag{14}$$

$$\mathcal{L}_{\text{abl}}(\hat{\mathbf{r}}) = \frac{1}{|D_{\text{harm}}|} \sum_{p \in D_{\text{harm}}} \text{CE}\big(f_{\text{abl}}(p; \hat{\mathbf{r}}),\ y_{\text{answer}}\big), \tag{15}$$

$$\mathcal{L}_{\text{retain}}(\hat{\mathbf{r}}) = \frac{1}{|D_{\text{safe}}|} \sum_{p \in D_{\text{safe}}} \text{KL}\big(f_{\text{abl}}(p; \hat{\mathbf{r}}) \,\|\, f_\theta(p)\big). \tag{16}$$

The total RDO loss objective is

$$\mathcal{L}_{\text{RDO}}(\hat{\mathbf{r}}) = \lambda_{\text{add}}\mathcal{L}_{\text{add}}(\hat{\mathbf{r}}) + \lambda_{\text{abl}}\mathcal{L}_{\text{abl}}(\hat{\mathbf{r}}) + \lambda_{\text{ret}}\mathcal{L}_{\text{retain}}(\hat{\mathbf{r}}). \tag{17}$$

RDO optimizes $\mathbf{r}$ by gradient descent through these modified passes and re-normalizes after each update, so the learned object is a direction.

Multi-directional refusal geometry: Refusal Cone Optimization (RCO). The single-direction RDO objective learns one vector, but refusal-related activations may not be captured by a single axis (Wollschläger et al., 2025; Pan et al., 2025). Refusal Cone Optimization (RCO) extends RDO by learning a region of directions in activation space for which the same intervention losses hold.

RCO maintains an orthonormal basis

$$B = [\mathbf{b}_1, \ldots, \mathbf{b}_N] \in \mathbb{R}^{d \times N}, \qquad B^\top B = I_N, \tag{18}$$

where each column $\mathbf{b}_i \in \mathbb{R}^d$ is a trainable direction. This basis defines an N-dimensional cone

$$\mathcal{R}_N = \Big\{ \sum_{i=1}^{N} \lambda_i \mathbf{b}_i \;\Big|\; \lambda_i \geq 0 \Big\} \setminus \{\mathbf{0}\}. \tag{19}$$

For any $\boldsymbol{\lambda} \in \mathbb{R}^N_{\geq 0}$, RCO forms the direction

$$\mathbf{r}_{\boldsymbol{\lambda}} = B\boldsymbol{\lambda}, \qquad \hat{\mathbf{r}}_{\boldsymbol{\lambda}} = \frac{\mathbf{r}_{\boldsymbol{\lambda}}}{\|\mathbf{r}_{\boldsymbol{\lambda}}\|_2}. \tag{20}$$

RCO reuses the RDO loss $\mathcal{L}_{\text{RDO}}(\cdot)$, now evaluated on sampled directions. Given sampled coefficients $\{\boldsymbol{\lambda}^{(m)}\}_{m=1}^M \subset \mathbb{R}^N_{\geq 0}$, define

$$\mathcal{L}_{\text{sample}}(B) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}_{\text{RDO}}\big(\hat{\mathbf{r}}_{\boldsymbol{\lambda}^{(m)}}\big). \tag{21}$$

To ensure that the basis vectors themselves satisfy the refusal properties, RCO additionally applies the same loss to each column of B:

$$\mathcal{L}_{\text{basis}}(B) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{\text{RDO}}(\mathbf{b}_i). \tag{22}$$

The full objective is

$$\mathcal{L}_{\text{RCO}}(B) = \mathcal{L}_{\text{sample}}(B) + \mathcal{L}_{\text{basis}}(B). \tag{23}$$

After every gradient update, the columns of B are orthonormalized (by Gram–Schmidt) to restore equation 18. This prevents co-linear directions and keeps the cone from degenerating into a single line.

C METHOD

C.1 STAGE 1: DISCOVERING THE REFUSAL GEOMETRY

Let f_θ be the base decoder-only Transformer, and let $\mathcal{D}_{\text{harm}}$ and $\mathcal{D}_{\text{safe}}$ be harmful and benign prompt sets. We run Refusal Cone Optimization (RCO) (Wollschläger et al., 2025) on the frozen model f_θ, using the additive and ablation interventions from Section B.2 applied at a fixed intervention site (a chosen layer and token position). RCO maintains an orthonormal basis

$$B = [\mathbf{b}_1, \ldots, \mathbf{b}_N] \in \mathbb{R}^{d \times N}, \qquad B^\top B = I_N,$$

and optimizes B so that for nonnegative coefficient vectors $\boldsymbol{\lambda} \in \mathbb{R}^N_{\geq 0}$, the normalized combination

$$\hat{\mathbf{r}}_{\boldsymbol{\lambda}} = \frac{B\boldsymbol{\lambda}}{\|B\boldsymbol{\lambda}\|_2}$$

satisfies the same three RDO losses (add, ablate, retain) as in equation 14–equation 16. While RCO defines a cone of directions via nonnegative combinations, our constraint in Stage 2 will use the associated linear span of the learned basis,

$$\mathcal{R} := \text{span}(B), \qquad P_{\mathcal{R}} := BB^\top, \qquad P_{\mathcal{R}^\perp} := I - BB^\top. \tag{24}$$

We refer to $\mathcal{R}$ as the refusal subspace induced by the learned cone basis.

C.2 Stage 2: Gated editing in $\mathcal{R}^{\perp}$

We focus on edits applied to attention-head components. Let $\mathbf{z}_t^{(l,i)} \in \mathbb{R}^d$ denote the activation of head i at layer l and token position t, where d is the head dimension. Following prior refusal-geometry interventions Wollschläger et al. (2025), we apply the decomposition at every token position, and restrict it to an early-layer prefix to avoid over-constraining representations near the unembedding: for a model with L layers, we intervene only for $l \in 0, \ldots, \lfloor 0.9L \rfloor - 1$. With the frozen cone, we decompose

$$\mathbf{z}_t^{(l,i)} = P_{\mathcal{R}} \mathbf{z}_t^{(l,i)} + P_{\mathcal{R}^{\perp}} \mathbf{z}_t^{(l,i)}.$$

We then apply JoLA-style gated hybrid edits (Lai et al., 2025) only to the orthogonal component. Let $\tilde{\mathbf{m}}^{(l,i)}, \tilde{\mathbf{a}}^{(l,i)} \in \mathbb{R}^d$ be unconstrained multiplicative and additive edit vectors and let $g_m^{(l,i)}, g_a^{(l,i)} \in [0, 1]$ be Hard–Concrete gates. Before the forward pass we project the parameters:

$$\mathbf{m}^{(l,i)} := P_{\mathcal{R}^{\perp}} \tilde{\mathbf{m}}^{(l,i)}, \qquad \mathbf{a}^{(l,i)} := P_{\mathcal{R}^{\perp}} \tilde{\mathbf{a}}^{(l,i)}. \tag{25}$$

To ensure the learned edit cannot reintroduce components along $\mathcal{R}$ (e.g., due to elementwise scaling), we additionally re-project the post-edit result back into $\mathcal{R}^{\perp}$. The edited activation is

$$\mathbf{z}_t^{(l,i)\prime} = P_{\mathcal{R}} \mathbf{z}_t^{(l,i)} + P_{\mathcal{R}^{\perp}} \Big(\big(\mathbf{1} + g_m^{(l,i)} \, \mathbf{m}^{(l,i)} \big) \odot \big(P_{\mathcal{R}^{\perp}} \mathbf{z}_t^{(l,i)} \big) + g_a^{(l,i)} \, \mathbf{a}^{(l,i)} \Big). \tag{26}$$

Thus, the refusal part $P_{\mathcal{R}} \mathbf{z}_t^{(l,i)}$ is always copied through unchanged, while the task-specific edit is constrained to lie in $\mathcal{R}^{\perp}$.

C.3 Overall Training Objective

Let $\mathcal{D} = \{(\mathbf{s}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$ be a supervised dataset, where $\mathbf{s}$ is the input token sequence and $\mathbf{y}$ are the target next-token labels. We train the edited model $f_{\theta,\phi}$ by minimizing the sum of the task loss and a sparsity regularizer on the Hard–Concrete gate parameters ψ:

$$\min_{\phi} \; \mathcal{L}_{\text{CE}}(\theta, \phi; \mathcal{D}) \; + \; \rho \, \Omega_{L_0}(\psi), \tag{27}$$

where $\mathcal{L}_{\text{CE}}$ is the standard next-token cross-entropy under the edited forward pass (i.e., using $\mathbf{z}_t^{(l,i)\prime}$ from equation 26), $\Omega_{L_0}(\psi)$ and $\rho \geq 0$ controls the sparsity–performance trade-off. We optimize only the intervention parameters $\phi = \{\mathbf{a}^{(l,i)}, \mathbf{m}^{(l,i)}, \psi_a^{(l,i)}, \psi_m^{(l,i)}\}$, keeping the base model parameters θ fixed.

D Experiment Configurations

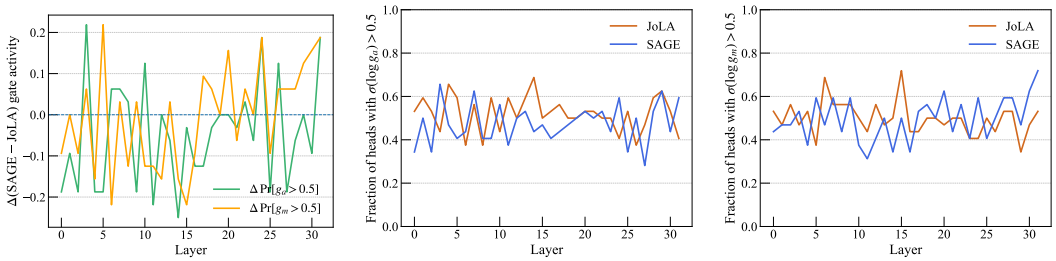
We fine-tune Llama-3.1-8B-Instruct in a low-resource regime with 200 training examples. Unless otherwise stated, we train for up to 20 epochs with batch size 16 using. We apply early stopping based on validation performance with patience 3. Table 2 lists method-specific hyperparameters.

Table 2: Method-specific training hyperparameters.

Method	Hyperparameter	Values
LoRA	learning rate	5×10^{-6}
LoRA	rank r	16
LoRA	α	32
LoRA	target modules	{q_proj, k_proj, v_proj, o_proj, up_proj, gate_proj, down_proj}
LoRA	LR scheduler	cosine
JoLA	applied module	attention
JoLA	learning rate	5×10^{-5}
JoLA	LR scheduler	cosine
JoLA	ρ	4×10^{-5}
JoLA	gate scheduler	expon
SAGE	cone dim N	{1,5}
SAGE	applied module	attention
SAGE	learning rate	5×10^{-5}
SAGE	LR scheduler	cosine
SAGE	ρ	4×10^{-5}
SAGE	gate scheduler	expon

E GATE ACTIVITY WITH AND WITHOUT THE REFUSAL-ORTHOGONALITY CONSTRAINT

We analyze how the refusal-orthogonality constraint affects the utilization of JoLA-style activation edits by examining layer-wise gate statistics. Figure 6 reports both absolute gate activity and relative changes induced by the constraint. Both methods are trained using the same settings except the refusal-orthogonality constraint.



(a) Change in gate activity (b) Active additive gates (c) Active multiplicative gates

Figure 6: **Gate-level effects of the refusal-orthogonality constraint. (a)** Difference in active-gate fraction per layer (SAGE − JoLA), showing a structured, sign-changing redistribution of gate activity across depth rather than uniform suppression. **(b–c)** Absolute fraction of heads with active additive (g_a) and multiplicative (g_m) gates, respectively. Across both gate types, SAGE retains substantial gate activity at most layers, comparable in magnitude to JoLA. Together, these plots show that safety improvements under SAGE arise from a depth-dependent reallocation of editing capacity, not from disabling the editor.

Insight 1: The refusal-orthogonality constraint does not suppress gate activity.
Relation to main result: rules out the interpretation that SAGE behaves as a weaker variant of unconstrained editing.

Across both additive and multiplicative edits, SAGE retains substantial gate activity throughout the network (Fig. 4–5). The fraction of heads with active gates remains comparable in magnitude to JoLA across most layers. Thus, SAGE preserves non-trivial adaptation capacity while achieving lower attack success rates in Table 1.

Insight 2: The refusal-orthogonality constraint induces non-uniform, layer-dependent changes in gate activity.
Relation to main result: shows that SAGE's differences from JoLA are not explained by a uniform, global weakening of the editor.

Figure 3 shows that the difference in active-gate fraction (SAGE − JoLA) varies in sign across layers, with some layers exhibiting increased gate usage and others decreased usage. This behavior is inconsistent with a uniform, depth-independent reduction in gate activity relative to JoLA, which would instead produce a near-constant negative shift across layers.