MITIGATING HARMFUL FINE-TUNING VIA TASK-AWARE DIRECTIONAL SAFETY REPAIR

Anonymous authors
Paper under double-blind review

ABSTRACT

Fine-tuning large language models on unvetted data can compromise their safety alignment, particularly when the training data contain harmful examples. However, preserving safety throughout harmful fine-tuning without hindering downstream task adaptation remains challenging. To address this challenge, we propose Task-Aware Directional Safety Repair (TDSR), a defense that integrates periodic safety repair into downstream fine-tuning. During each safety repair, TDSR uses gradients of the safety and downstream task losses to identify candidate directions along which parameter updates strongly affect safety losses while causing relatively small changes in task losses. It then selects the candidate most closely aligned with the direction of steepest decrease in the current safety loss and projects the safety descent gradient onto the selected candidate direction to update the model. Extensive experiments across various models and downstream tasks show that TDSR preserves safety alignment while maintaining task performance, outperforming existing baselines. For example, on Mistral-7B-v0.3, TDSR reduces the mean harmful-response rate by 15.62 percentage points and improves mean task performance by 11.30 percentage points across three tasks, relative to the strongest baselines for safety and downstream task performance, respectively.

1 INTRODUCTION

Downstream fine-tuning enables large language models (LLMs) to acquire task-specific capabilities, but can also weaken their safety alignment. When harmful examples are mixed into the training data, fine-tuning can compromise this alignment and make the model more likely to comply with malicious requests (Qi et al., 2024; Huang et al., 2024a). Such safety degradation can occur even when the fine-tuning data are entirely benign (Qi et al., 2024). These risks motivate defenses that preserve safety throughout fine-tuning while allowing effective downstream task adaptation.

Existing defenses address safety degradation during fine-tuning by constraining harmful updates (Zhang et al., 2026; Lu et al., 2026), strengthening alignment robustness (Huang et al., 2025b; Nguyen et al., 2026), or restoring safety alignment (Huang et al., 2024a; 2025a; Wang et al., 2026b). However, update constraints may also restrict benign task learning. Strengthening alignment can incur utility costs, and its protection may weaken as harmful-data ratios or training duration increase (Wang et al., 2026b; Nguyen et al., 2026). One-time repair may not reverse accumulated safety degradation (Huang et al., 2025a; Wang et al., 2026b), whereas repeated repairs may disrupt task learning (Huang et al., 2024a). Thus, preserving safety alignment throughout fine-tuning without hindering downstream task adaptation remains challenging.

To address this challenge, we propose **T**ask-Aware **D**irectional **S**afety **R**epair (TDSR), which periodically updates model parameters during fine-tuning along directions that restore safety alignment while preserving downstream task performance. Specifically, we use losses computed on the defender's safety and task reference datasets to assess safety and task performance, respectively, and therefore seek update directions that strongly affect safety losses while having relatively little effect on task losses. We define safety and task sensitivity to quantify how strongly these losses respond to small parameter changes along a given direction. Using gradients of these reference losses, TDSR searches for candidate directions by maximizing a regularized ratio of safety sensitivity to task sensitivity. Then, TDSR selects the candidate direction most closely aligned with the direction of steepest decrease in the current safety loss. It projects the safety descent gradient onto this direction to update

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.

the model. The direction search exploits the low rank of the reference gradient matrices to avoid constructing matrices at the full parameter dimension. After repair, downstream training resumes from the updated parameters.

Experiments across five LLMs show that TDSR outperforms the evaluated baselines in preserving safety alignment while maintaining performance on mathematical reasoning, instruction following, and code generation. We compare six baselines on four models and three on the fifth. For instance, on Mistral-7B-v0.3 with AlpacaEval, TDSR reduces the harmful-response rate by 17.43 percentage points and improves the win rate by 22.86 percentage points relative to the strongest baseline. Further experiments support its robustness to longer fine-tuning, higher harmful-data ratios, and two adaptive attacks, while ablation studies demonstrate the benefits of the repair direction strategy and its components. Through mechanism analysis, we provide evidence that TDSR offers a distinct route to safety repair by forming new activation patterns for harmful inputs rather than simply preserving those of the original aligned model.

Our contributions are summarized as follows:

- We propose TDSR, a periodic safety repair method that identifies update directions by contrasting safety and task loss sensitivities, enabling safety recovery during fine-tuning while limiting interference with downstream task learning.
- Our mechanism analysis provides evidence that TDSR offers a distinct route to safety fine-tuning by forming new activation patterns for harmful inputs, rather than simply preserving those of the original aligned model.
- We demonstrate the effectiveness and robustness of TDSR in preserving safety alignment while maintaining downstream task performance through extensive experiments across five models and three tasks, including baseline comparisons, ablation studies, and evaluations under varied fine-tuning conditions and adaptive attacks.

2 Background & Related Work

2.1 Downstream Fine-Tuning

Downstream fine-tuning adapts language models to tasks such as text classification and mathematical reasoning. Supervised fine-tuning maximizes the likelihood of target responses to training prompts (Wei et al., 2022; Sanh et al., 2022). However, harmful fine-tuning can weaken safety safeguards in open-weight and API-served models (Yang et al., 2023; Lermen & Rogers-Smith, 2024; Zhan et al., 2024). Safety can also degrade under entirely benign customization (Qi et al., 2024; Guan et al., 2025).

2.2 Defenses against Harmful Fine-Tuning

We group defenses against harmful fine-tuning by their primary mechanisms into three categories: constraining harmful updates, enhancing alignment robustness, and restoring safety alignment.

Constraining Harmful Updates. These defenses restrict harmful learning during fine-tuning. Shen et al. (2025); Hu et al. (2026); Peng et al. (2026); Wang et al. (2026a) control training-example or response-token contributions through selection or reweighting. Yang et al. (2026); Li et al. (2025); Zhang et al. (2026) constrain parameter updates or adapter-induced features to preserve alignment. Other methods constrain initial-token probabilities (Qi et al., 2025) or anchor the final hidden states of harmful prompts to those of the aligned model (Lu et al., 2026).

Enhancing Alignment Robustness. These defenses strengthen alignment before downstream fine-tuning. Huang et al. (2024b); Liu et al. (2025); Tang et al. (2026); Chen et al. (2025) use adversarial perturbations, while Rosati et al. (2024) regularize harmful representations toward noise. Huang et al. (2025b); Tamirisa et al. (2025) optimize resistance to simulated harmful updates, and Nguyen et al. (2026) flatten harmful-loss landscapes and reweight downstream data. Other approaches fortify alignment through curated clean data (Liu et al., 2024) or training-free safety-subspace extrapolation (Perin et al., 2025). Yi et al. (2026) instead embed a trap designed to degrade

general capability under persistent harmful adaptation while remaining dormant during benign fine-tuning.

Restoring Safety Alignment. Repair-based defenses restore safety during or after fine-tuning. Huang et al. (2024a) alternate task training and safety alignment under proximal regularization. Hsu et al. (2024); Yi et al. (2024); Lu et al. (2025) use parameter projection, subspace fusion, or selective adjustment of parameter changes. Huang et al. (2025a); Yi et al. (2025) prune weights associated with harmful generation or replace safety-critical neurons using a reference model. Wang et al. (2026b); Li et al. (2026) learn post-tuning parameter perturbations or reusable patches, while Zhu et al. (2024) transfer safety-related activation directions from the original model for inference-time intervention.

Limitations of Existing Defenses. Existing defenses still face challenges in maintaining safety while preserving downstream utility. Constraints intended to suppress harmful updates may also restrict the directions needed for benign task adaptation when the two overlap. Robustness-enhancing alignment can incur utility costs, and its protection may weaken as the harmful-data ratio or the extent of fine-tuning increases (Wang et al., 2026b; Nguyen et al., 2026). Repair-based defenses may struggle to reverse accumulated safety degradation with a single intervention (Huang et al., 2025a; Wang et al., 2026b), while repeated corrections risk disrupting task learning (Huang et al., 2024a).

3 Threat Model

Setting. We study defenses against harmful fine-tuning from the perspective of a service provider that adapts a safety-aligned language model M, parameterized by $\boldsymbol{\theta}_0$, to a user's downstream task. The user collects and submits a dataset $\mathbb{D}_{\mathrm{ft}}$ of prompts and corresponding responses. The provider applies a fine-tuning procedure $\mathcal{F}$ to obtain the adapted parameters

$$\boldsymbol{\theta}_{\mathrm{ft}} = \mathcal{F}(\mathbb{D}_{\mathrm{ft}}, \boldsymbol{\theta}_0). \tag{1}$$

3.1 Attacker Goals and Capabilities

The attacker aims to weaken safety alignment by injecting selected harmful examples into otherwise benign data sources. Users may inadvertently include these examples in $\mathbb{D}_{\mathrm{ft}}$ when collecting training data. We denote the harmful subset of $\mathbb{D}_{\mathrm{ft}}$ by $\mathbb{D}_{\mathrm{harm}}$, which contains harmful prompts paired with unsafe responses, and define the harmful-data ratio as $p = |\mathbb{D}_{\mathrm{harm}}|/|\mathbb{D}_{\mathrm{ft}}|$. The attacker is limited to data contamination and cannot modify the provider's training procedure or trusted reference data.

3.2 Defender Goals and Capabilities

The provider acts as the defender, aiming to maintain safe responses to malicious queries while retaining performance gains on benign downstream tasks. We assume the defender has full control over fine-tuning and can access model parameters, compute gradients, modify parameter updates, and apply safety corrections before releasing the adapted model.

Defender's Knowledge. Following prior work (Gu et al., 2025; Li et al., 2026), we assume the defender has access to two small reference sets to guide the defense. The safety set $\mathbb{D}_s$ contains harmful prompts paired with safe responses, while the task set $\mathbb{D}_t$ contains benign examples of the downstream task. Both are disjoint from the submitted fine-tuning data and evaluation data.

4 Methodology

We propose **T**ask-Aware **D**irectional **S**afety **R**epair (TDSR), which integrates periodic safety repair into downstream fine-tuning to preserve safety alignment and task utility. During each repair, TDSR seeks directions that strongly affect safety losses while limiting task-loss changes, then selects the one expected to reduce the current safety loss most to update the model.

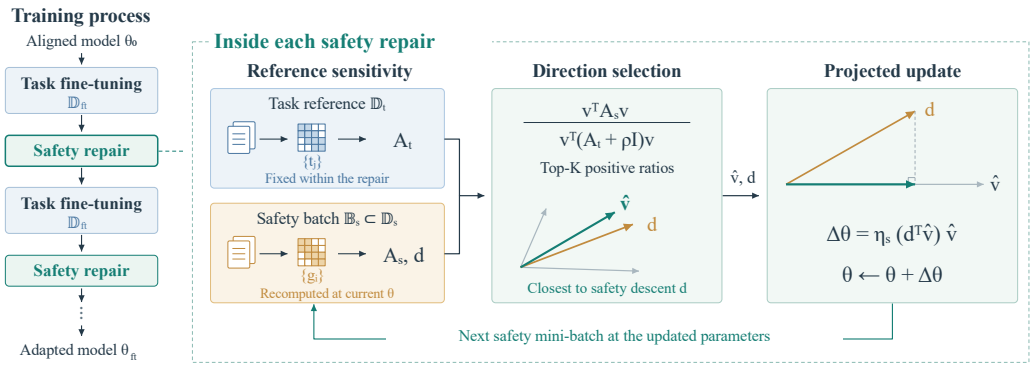

Figure 1: Overview of TDSR. Downstream fine-tuning alternates with periodic safety repair. The dashed box shows candidate direction search using safety and task sensitivities, selection by alignment with the safety descent gradient, and the projected update.

4.1 Overall Framework

Figure 1 summarizes the procedure, with Algorithm 1 in Appendix A. Starting from aligned parameters $\boldsymbol{\theta}_0$, the defender trains on $\mathbb{D}_{\text{ft}}$ using the standard task optimizer. At scheduled steps, training pauses for repair using safety reference sets $\mathbb{D}_s$ and task reference sets, maintained by the defender. After repair, training then resumes from the repaired parameters with its optimizer state retained.

For each repair, TDSR first recomputes task gradients on $\mathbb{D}_t$ and keeps them fixed throughout the repair. For each successive mini-batch from $\mathbb{D}_s$, it computes safety gradients at the current parameters and uses both gradient sets to identify candidate directions (Section 4.2). It then updates the model along the candidate expected to reduce the mini-batch safety loss most (Section 4.3).

4.2 Sensitivity-Guided Direction Search

The key to a repair event is to identify update directions along which small parameter changes strongly affect safety losses but have relatively little effect on task losses. In this subsection, we define directional loss sensitivity to quantify these effects, then derive an efficient procedure for finding candidate repair directions.

Definition 1 (Directional Loss Sensitivity) Let P denote the data distribution of the task of interest, such as safety alignment or logical reasoning. The directional loss sensitivity of this task at model parameters $\boldsymbol{\theta} \in \mathbb{R}^m$ along a unit direction $\boldsymbol{v} \in \mathbb{R}^m$ is defined as

$$\mathcal{S}_P(\boldsymbol{v};\boldsymbol{\theta}) = \mathbb{E}_{(\mathrm{x},\mathrm{y})\sim P}\left[\left(\boldsymbol{v}^\top \nabla_{\boldsymbol{\theta}} \ell(\mathrm{x},\mathrm{y};\boldsymbol{\theta})\right)^2\right], \qquad \|\boldsymbol{v}\|_2 = 1. \tag{2}$$

Here, (x, y) denotes a sample drawn from P with prompt x and target response y, and $\ell(x, y; \boldsymbol{\theta})$ is the corresponding differentiable loss. The inner product $\boldsymbol{v}^\top \nabla_{\boldsymbol{\theta}} \ell(x, y; \boldsymbol{\theta})$ measures the instantaneous rate of change in the loss on (x, y) as the parameters move along $\boldsymbol{v}$:

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon}\ell(x, y; \boldsymbol{\theta} + \epsilon\boldsymbol{v})\bigg|_{\epsilon=0} = \boldsymbol{v}^\top \nabla_{\boldsymbol{\theta}} \ell(x, y; \boldsymbol{\theta}).$$

Replacing $\boldsymbol{v}$ with $-\boldsymbol{v}$ reverses the sign of $\boldsymbol{v}^\top \nabla_{\boldsymbol{\theta}} \ell(x, y; \boldsymbol{\theta})$ while preserving its magnitude. To capture this magnitude without distinguishing loss increases from decreases, we square the directional derivative before taking its expectation. Thus, $\mathcal{S}_P(\boldsymbol{v}; \boldsymbol{\theta})$ quantifies how strongly losses on the task represented by P respond to small parameter updates along $\boldsymbol{v}$.

For fixed model parameters, the sensitivity in Definition 1 is a quadratic function of $\boldsymbol{v}$:

$$\mathcal{S}_P(\boldsymbol{v};\boldsymbol{\theta}) = \boldsymbol{v}^\top \mathbb{E}_{(\mathrm{x},\mathrm{y})\sim P}\left[\nabla_{\boldsymbol{\theta}} \ell(\mathrm{x},\mathrm{y};\boldsymbol{\theta})\left(\nabla_{\boldsymbol{\theta}} \ell(\mathrm{x},\mathrm{y};\boldsymbol{\theta})\right)^\top\right] \boldsymbol{v}.$$

The matrix in this expression is independent of $\boldsymbol{v}$ and can be estimated from reference gradients before selecting a direction. In practice, TDSR constructs the task and safety sensitivity functions of $\boldsymbol{v}$ using gradients from $\mathbb{D}_t$ and $\mathbb{D}_s$, respectively.

Task Sensitivity Function. At the start of each repair event, TDSR computes gradients on the entire task reference set $\mathbb{D}_t$ to construct the task sensitivity function and keeps this function fixed throughout the event. Let $\boldsymbol{\theta}^{\text{ref}}$ denote the parameters at the start of a repair event. For each task reference example $(x_j^t, y_j^t) \in \mathbb{D}_t$, we compute $\boldsymbol{t}_j = \nabla_{\boldsymbol{\theta}} \ell(x_j^t, y_j^t; \boldsymbol{\theta})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{\text{ref}}}$ and define

$$\boldsymbol{A}_t = \frac{1}{|\mathbb{D}_t| s_t^2} \sum_{j=1}^{|\mathbb{D}_t|} \boldsymbol{t}_j \boldsymbol{t}_j^\top, \qquad s_t = \text{median}_j \, \|\boldsymbol{t}_j\|_2. \tag{3}$$

Here, s_t is the median norm of the task gradients computed on $\mathbb{D}_t$. For a unit direction $\boldsymbol{v}$, $\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v}$ estimates the normalized sensitivity of task reference losses to small parameter changes along $\boldsymbol{v}$.

Safety Sensitivity Function. TDSR recomputes the safety sensitivity function once per mini-batch from $\mathbb{D}_s$ using the current model parameters. Let $\mathbb{B}_s^{(r)} = \{(x_i, y_i^{\text{safe}})\}_{i=1}^{|\mathbb{B}_s^{(r)}|}$ denote the r-th mini-batch from $\mathbb{D}_s$ within a repair event, and $\boldsymbol{\theta}^{(r)}$ the current model parameters. For this mini-batch, we compute the gradients $\boldsymbol{g}_i = \nabla_{\boldsymbol{\theta}} \ell(x_i, y_i^{\text{safe}}; \boldsymbol{\theta})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(r)}}$ and define

$$\boldsymbol{A}_s = \frac{1}{|\mathbb{B}_s^{(r)}| s_s^2} \sum_{i=1}^{|\mathbb{B}_s^{(r)}|} \boldsymbol{g}_i \boldsymbol{g}_i^\top, \qquad s_s = \text{median}_i \, \|\boldsymbol{g}_i\|_2. \tag{4}$$

Here, s_s is the median norm of the safety gradients in the current mini-batch. For a unit direction $\boldsymbol{v}$, $\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v}$ estimates the normalized sensitivity of safety reference losses to small parameter changes along $\boldsymbol{v}$.

Candidate Direction Search. For each mini-batch from $\mathbb{D}_s$, once the task and safety sensitivity functions are available, TDSR searches for candidate directions with high safety sensitivity relative to task sensitivity. To identify such directions, we use a regularized ratio of safety sensitivity to task sensitivity as the search objective, favoring larger ratios:

$$R(\boldsymbol{v}) = \frac{\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v}}{\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v} + \rho \|\boldsymbol{v}\|_2^2} = \frac{\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v}}{\boldsymbol{v}^\top \boldsymbol{B}_t \boldsymbol{v}}, \qquad \boldsymbol{v} \neq \boldsymbol{0}. \tag{5}$$

Here, we use a regularized version of $\boldsymbol{A}_t$, denoted $\boldsymbol{B}_t = \boldsymbol{A}_t + \rho \boldsymbol{I}$ with $\rho > 0$, to ensure a positive denominator even for directions with zero empirical task sensitivity (see Appendix B for a detailed analysis). To obtain candidate repair directions with large values of $R(\boldsymbol{v})$, we solve the generalized eigenvalue problem arising from its stationary condition $\nabla_{\boldsymbol{v}} R(\boldsymbol{v}) = \boldsymbol{0}$. The resulting eigenvectors $\boldsymbol{v}_k$ satisfy

$$\boldsymbol{A}_s \boldsymbol{v}_k = R(\boldsymbol{v}_k) \boldsymbol{B}_t \boldsymbol{v}_k. \tag{6}$$

We retain up to K eigenvectors with the largest positive eigenvalues, which equal their sensitivity ratios $R(\boldsymbol{v}_k)$, for subsequent selection in Section 4.3.

Efficient Computation. TDSR reduces the generalized eigenvalue problem to the safety mini-batch dimension while preserving its positive eigenvalues. Let $n_s = |\mathbb{B}_s^{(r)}|$ and $n_t = |\mathbb{D}_t|$, and stack the scaled gradients as

$$\boldsymbol{X} = \frac{1}{s_s \sqrt{n_s}} [\boldsymbol{g}_1, \ldots, \boldsymbol{g}_{n_s}]^\top \in \mathbb{R}^{n_s \times m}, \qquad \boldsymbol{Y} = \frac{1}{s_t \sqrt{n_t}} [\boldsymbol{t}_1, \ldots, \boldsymbol{t}_{n_t}]^\top \in \mathbb{R}^{n_t \times m}. \tag{7}$$

Then $\boldsymbol{A}_s = \boldsymbol{X}^\top \boldsymbol{X}$ and $\boldsymbol{B}_t = \rho \boldsymbol{I} + \boldsymbol{Y}^\top \boldsymbol{Y}$.

Proposition 1 (Exact Reduced-Space Direction Search) *For $\rho > 0$, define*

$$\boldsymbol{Q} = \boldsymbol{X} \boldsymbol{B}_t^{-1} \boldsymbol{X}^\top \in \mathbb{R}^{n_s \times n_s}. \tag{8}$$

The positive eigenvalues of $\boldsymbol{Q}$ coincide with the positive generalized eigenvalues of $(\boldsymbol{A}_s, \boldsymbol{B}_t)$, including multiplicities. For any eigenpair $\boldsymbol{Q}\boldsymbol{a} = \lambda \boldsymbol{a}$ with $\lambda > 0$, the direction

$$\boldsymbol{v} = \boldsymbol{B}_t^{-1} \boldsymbol{X}^\top \boldsymbol{a} = \frac{1}{\rho} \boldsymbol{X}^\top \boldsymbol{a} - \frac{1}{\rho^2} \boldsymbol{Y}^\top \left(\boldsymbol{I}_{n_t} + \frac{1}{\rho} \boldsymbol{Y} \boldsymbol{Y}^\top \right)^{-1} \boldsymbol{Y} \boldsymbol{X}^\top \boldsymbol{a} \tag{9}$$

is nonzero and satisfies $\boldsymbol{A}_s \boldsymbol{v} = \lambda \boldsymbol{B}_t \boldsymbol{v}$ and $R(\boldsymbol{v}) = \lambda$.

Accordingly, TDSR solves for up to K eigenvectors of $\boldsymbol{Q}$ with the largest positive eigenvalues and recovers the corresponding candidate directions using Equation 9. Section 4.3 describes their selection for the repair update. Products with $\boldsymbol{B}_t^{-1}$ require only an n_t-dimensional linear solve, without constructing any $m \times m$ matrix; Appendix C provides the proof and computational details.

4.3 Descent-Aligned Safety Repair

Given the candidates from Section 4.2, TDSR uses the current safety-loss gradient to select the repair direction and determine the update magnitude.

Descent-Aligned Selection. Since the leading sensitivity-ratio directions have only small effects on task losses, TDSR prioritizes safety recovery by selecting the candidate most closely aligned with safety descent. We obtain the direction of steepest decrease in the current mini-batch safety loss from

$$\mathcal{L}_s(\boldsymbol{\theta}) = \frac{1}{|\mathbb{B}_s^{(r)}|} \sum_{i=1}^{|\mathbb{B}_s^{(r)}|} \ell(x_i, y_i^{\text{safe}}; \boldsymbol{\theta}), \qquad \boldsymbol{d} = -\nabla_{\boldsymbol{\theta}} \mathcal{L}_s(\boldsymbol{\theta}) = -\frac{1}{|\mathbb{B}_s^{(r)}|} \sum_{i=1}^{|\mathbb{B}_s^{(r)}|} \boldsymbol{g}_i. \tag{10}$$

Each candidate is normalized to $\boldsymbol{u}_k = \boldsymbol{v}_k / \|\boldsymbol{v}_k\|_2$ and oriented so that $\boldsymbol{d}^\top \boldsymbol{u}_k \geq 0$. We select the candidate most closely aligned with $\boldsymbol{d}$:

$$k^* = \arg\max_k \frac{\boldsymbol{d}^\top \boldsymbol{u}_k}{\|\boldsymbol{d}\|_2}, \qquad \hat{\boldsymbol{v}} = \boldsymbol{u}_{k^*}. \tag{11}$$

Repair Update. TDSR projects $\boldsymbol{d}$ onto the selected unit direction $\hat{\boldsymbol{v}}$ to update the model with step size $\eta_s > 0$:

$$\Delta\boldsymbol{\theta} = \eta_s (\boldsymbol{d}^\top \hat{\boldsymbol{v}}) \hat{\boldsymbol{v}}, \qquad \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}. \tag{12}$$

Since $\boldsymbol{d}^\top \hat{\boldsymbol{v}} = \|\boldsymbol{d}\|_2 \cos \angle(\boldsymbol{d}, \hat{\boldsymbol{v}})$, the update magnitude depends on both the safety-gradient magnitude and its alignment with the selected direction. Every nonzero update is a descent direction for the current mini-batch safety loss and decreases it for a sufficiently small step size (Appendix C.2).

5 Evaluation

5.1 Experimental Setup

Models and Datasets. We evaluate Qwen2.5-7B-Instruct (Qwen et al., 2025), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Gemma-2-9B-IT (Team et al., 2024), and Mistral-7B-Instruct-v0.3 (Mistral AI, 2024). Fine-tuning sets contain 1,000 examples each for GSM8K (Cobbe et al., 2021) (mathematical reasoning) and Magicoder (Wei et al., 2024b) (code generation), and 700 for AlpacaEval (Li et al., 2023) (instruction following). Following prior work (Huang et al., 2025b; Nguyen et al., 2026), we mix benign task data with harmful examples randomly sampled from BeaverTails (Ji et al., 2023) at a default harmful ratio $p = 0.2$. The defender's reference sets, disjoint from fine-tuning and evaluation data, contain 200 randomly sampled harmful BeaverTails prompts paired with safe responses ($\mathbb{D}_s$) and 64 benign examples per task dataset ($\mathbb{D}_t$).

Baselines. We compare TDSR with standard SFT without a safety defense, LISA (Huang et al., 2024a), Booster (Huang et al., 2025b), OASIS (Tang et al., 2026), SBR (Lu et al., 2026), and GuardSpace (Zhang et al., 2026).

Evaluation Metrics. *Harmful Score (HS)* is the percentage of responses classified as harmful by Beaver-Dam (Ji et al., 2023) on held-out harmful prompts. *Fine-tuning Accuracy (FA)* is answer accuracy on GSM8K, pairwise win rate on AlpacaEval, or pass@1 on HumanEval+ using EvalPlus (Liu et al., 2023). Lower HS and higher FA indicate better performance.

Training Configuration. We use LoRA (Hu et al., 2022) with rank $r = 8$ for downstream fine-tuning, as in prior work (Huang et al., 2025b; Lu et al., 2026), and train for 20 epochs. TDSR performs two repairs per epoch. Appendix D details data construction, evaluation, and hyperparameters. Appendices E and H analyze sensitivity and efficiency.

Table 1: Results (%) across three downstream tasks and four models at $p = 0.2$. Average is the mean over four models. For each dataset and metric column, the best result is **bold with a green background**, while the second best is **bold with a gray background**.

Dataset	Method	Harmful Score ↓					Fine-tuning Accuracy ↑				
		Qwen 2.5-7B	Llama 3.1-8B	Gemma 2-9B	Mistral 7B-v0.3	Avg.	Qwen 2.5-7B	Llama 3.1-8B	Gemma 2-9B	Mistral 7B-v0.3	Avg.
GSM8K	Vanilla SFT	73.92	67.85	81.57	81.29	76.16	77.71	79.00	75.51	**43.06**	**68.82**
	LISA	13.57	30.86	14.29	43.00	25.43	**78.32**	73.46	75.74	42.30	67.46
	SBR	6.29	**4.71**	9.71	**27.57**	**12.07**	77.26	**79.53**	76.42	35.56	67.19
	OASIS	7.86	33.29	21.00	44.14	26.57	76.65	67.02	68.84	39.58	63.02
	Booster	**6.00**	35.00	6.29	29.29	19.15	70.36	69.14	51.71	40.86	58.02
	GuardSpace	23.43	37.71	**4.14**	41.43	26.68	76.72	72.33	**80.36**	40.26	67.42
	TDSR	**5.86**	**2.57**	**2.71**	**9.71**	**5.21**	**83.77**	**79.68**	**77.94**	**43.21**	**71.15**
AlpacaEval	Vanilla SFT	41.29	41.86	42.00	45.43	42.65	47.14	46.19	43.81	40.48	44.41
	LISA	27.57	28.86	12.86	40.71	27.50	**54.29**	47.62	**57.14**	40.00	49.76
	SBR	**8.00**	**6.57**	15.14	**30.57**	**15.07**	53.52	**54.76**	48.57	**48.57**	**51.36**
	OASIS	9.71	36.71	32.43	43.57	30.61	47.17	38.10	42.38	35.23	40.72
	Booster	20.29	31.71	**6.00**	38.29	24.07	43.81	44.29	30.00	38.57	39.17
	GuardSpace	16.71	35.43	10.86	40.71	25.93	**55.24**	43.33	54.28	47.14	50.00
	TDSR	**6.57**	**2.43**	**2.71**	**13.14**	**6.21**	53.33	**59.05**	**56.19**	**71.43**	**60.00**
Magicoder	Vanilla SFT	48.29	59.15	46.86	45.14	49.86	70.12	40.14	53.66	28.05	47.99
	LISA	9.71	14.14	8.14	38.43	17.61	70.73	55.49	**58.53**	27.44	53.05
	SBR	**4.43**	5.14	11.43	**24.14**	**11.29**	**79.88**	54.27	54.88	26.27	**53.83**
	OASIS	26.86	38.00	14.29	43.43	30.65	68.29	53.66	51.83	23.78	49.39
	Booster	6.71	41.71	**6.43**	36.43	22.82	65.24	56.10	49.39	**29.88**	50.15
	GuardSpace	8.29	**4.29**	7.57	30.57	12.68	73.78	**57.32**	55.49	26.27	53.22
	TDSR	**4.57**	**3.57**	**3.14**	**12.57**	**5.96**	**81.10**	**60.37**	**57.32**	**32.93**	**57.93**

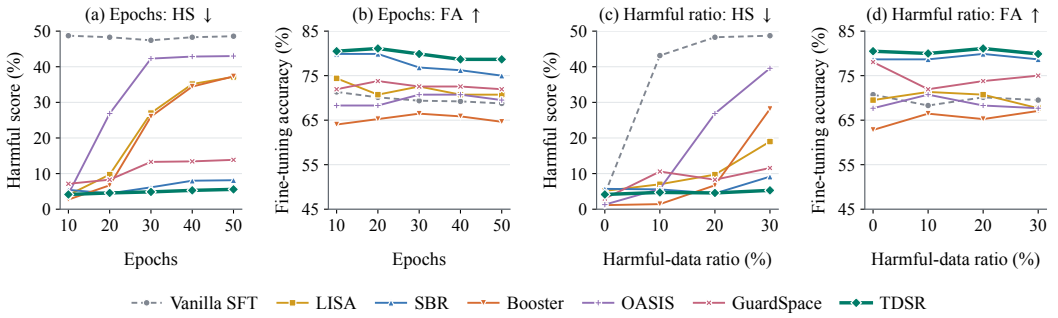

Figure 2: Robustness evaluation results for Qwen2.5-7B-Instruct fine-tuned on Magicoder. Panels (a,b) vary the fine-tuning duration at a harmful-data ratio of $p = 0.2$, while panels (c,d) vary the harmful-data ratio at 20 epochs. Each pair shows HS on the left and FA on the right.

5.2 Effectiveness and Robustness

Main Evaluation. Table 1 compares TDSR with six baselines across four models and three tasks at $p = 0.2$. Averaged across the four models, TDSR achieves the lowest HS and highest FA on each task, lowering HS by 5.32–8.86 percentage points relative to the strongest baseline on each task. Across 12 combinations of models and tasks, it achieves the lowest HS in 11 and highest FA in eight. The HS exception, Qwen2.5-7B-Instruct on Magicoder, is only 0.14 percentage points above the best result. Appendices F and I report additional safety tests and Qwen3.8-27B results.

Table 2: Ablation results (%) on Qwen2.5-7B-Instruct with Magicoder at $p = 0.2$. Orange values in parentheses show HS increases and FA decreases relative to the full method, in percentage points. The full method is highlighted in green, with its results in bold.

Setting	Harmful Score ↓	Fine-tuning Accuracy ↑
Vanilla SFT	48.29 (↑ 43.72)	70.12 (↓ 10.98)
Plain safety SFT repair	6.43 (↑ 1.86)	68.99 (↓ 12.11)
LISA (same repair frequency)	34.86 (↑ 30.29)	67.07 (↓ 14.03)
Task-orthogonal repair	12.29 (↑ 7.72)	79.23 (↓ 1.87)
Without task sensitivity	6.14 (↑ 1.57)	69.98 (↓ 11.12)
Highest-ratio direction only	8.00 (↑ 3.43)	79.43 (↓ 1.67)
TDSR (full)	**4.57**	**81.10**

Robustness. On Qwen2.5-7B-Instruct with Magicoder, we vary fine-tuning from 10 to 50 epochs at $p = 0.2$ and the harmful-data ratio from 0 to 0.3 at 20 epochs (Appendix D.4). TDSR achieves the highest FA at every setting, with HS of 4.14–5.57% across durations and 4.14–5.29% across ratios (Figure 2). It achieves the lowest HS at 30, 40, and 50 epochs and at $p = 0.3$. At 50 epochs, HS is 2.57 percentage points lower and FA 3.66 points higher than the strongest baseline for each metric. At $p = 0.3$, these improvements are 3.85 and 1.22 percentage points, respectively. Appendix G reports results on Mistral-7B-v0.3.

5.3 Ablation Study

We evaluate the repair direction strategy of TDSR and its components, comparing alternative directions under the same schedule of two repairs per epoch. We then ablate safety sensitivity, task sensitivity, and candidate selection by alignment with the steepest descent direction of the current safety loss. We conduct this evaluation using Qwen2.5-7B-Instruct on Magicoder.

Comparison with Alternative Repair Direction Strategies. Table 2 shows that TDSR achieves a better balance between safety and utility. Task-orthogonal repair leaves HS at 12.29%, plain safety SFT repair reaches only 68.99% FA, and LISA is weaker on both metrics. Only TDSR achieves both the lowest HS and the highest FA among these repair strategies.

Effectiveness of Key Components. The ablation results in Table 2 show that removing components of repair direction selection weakens both safety recovery and task preservation. For example, removing task sensitivity reduces FA by 11.12 percentage points. Retaining both sensitivities and candidate selection gives TDSR the best balance between safety and utility among these variants.

5.4 Adaptive Attacks

We evaluate TDSR against two adaptive attacks that aim to make safety repair interfere with task performance by increasing task sensitivity along harmful-gradient directions. The attacker knows the defense and its 64 task reference examples, but cannot modify the reference sets or defense hyperparameters. For each harmful example, we average this sensitivity over 20 checkpoints from the standard TDSR run. The first attack, *gradient selection*, retains the 200 harmful candidates with the highest average sensitivity. The second, *gradient suffix*, optimizes a shared 16-token suffix appended to harmful training inputs to increase this average sensitivity. We construct both attacks independently for each of the four models, train from the original initialization on Magicoder for 20 epochs at $p = 0.2$, and evaluate without the suffix (Appendix D.5).

Results. As shown in Table 3, HS remains at most 5.86% on Qwen2.5, Llama3.1, and Gemma2, and at most 12.29% on Mistral under both attacks. Compared with the standard TDSR results in Table 1, HS increases by at most 1.29 percentage points, and FA decreases by at most 1.82 percentage points across both attacks and all four models. These results support the resilience of TDSR to the two tested adaptive attacks while retaining downstream task performance.

Table 3: Results (%) of TDSR under two adaptive attacks on Magicoder at $p = 0.2$ after 20 epochs.

Attack	Qwen2.5-7B		Llama-3.1-8B		Gemma-2-9B		Mistral-7B-v0.3	
	HS ↓	FA ↑	HS ↓	FA ↑	HS ↓	FA ↑	HS ↓	FA ↑
Gradient selection	4.57	79.28	3.71	59.15	2.86	60.36	12.14	33.54
Gradient suffix	5.86	81.71	4.00	61.59	3.57	58.53	12.29	32.31

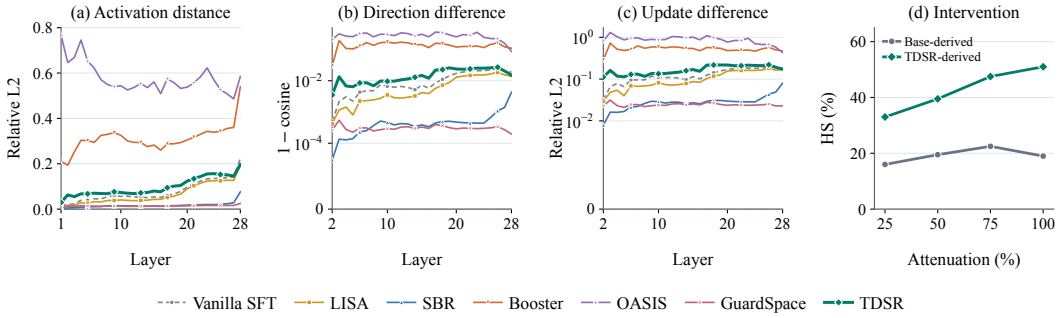

Figure 3: Mechanism analysis on Qwen2.5-7B-Instruct with Magicoder. Mean distances from Base over 700 harmful prompts: (a) relative activation distance and (b,c) cosine and relative L2 distances between consecutive-layer activation changes. Panel (d) shows HS after attenuating activation components along Base- or TDSR-derived directions in the same TDSR model (200 harmful prompts). Panels (b,c) use symmetric logarithmic scales.

5.5 Mechanism Analysis

To understand how TDSR restores safety alignment, we compare the model's activations with those of the original aligned model (Base) to examine its internal response patterns to harmful inputs.

Representation Changes. Using 700 harmful prompts, we compare representations from Qwen2.5-7B-Instruct fine-tuned on Magicoder. Figure 3(a) measures activation differences from Base at each layer. Panels (b,c) compare changes between consecutive layers with those of Base, measuring differences in direction and in both direction and magnitude, respectively. See Appendix D.6 for details. Except for Booster and OASIS, which strengthen alignment before downstream fine-tuning, TDSR generally deviates further from Base than the other methods, including vanilla SFT, which exhibits representation drift and a high HS. These observations suggest that TDSR reorganizes its internal activation patterns on harmful inputs during task-aware safety repair.

Direction Interventions. To examine this interpretation, we attenuate activation components along Base- or TDSR-derived directions in the same TDSR model and evaluate both interventions on 200 harmful prompts. Attenuating TDSR-derived directions yields higher HS than attenuating Base-derived directions at every tested strength, reaching 51% versus 19% at full attenuation, a gap of 32 percentage points (Figure 3(d)). These results suggest that TDSR restores safety by reorganizing internal activation patterns on harmful inputs, rather than simply preserving those of Base.

6 Conclusion

In this paper, we propose TDSR, a defense that mitigates harmful fine-tuning through periodic safety repair along task-aware directions. Specifically, by contrasting the directional sensitivities of safety and task losses, TDSR identifies repair directions that favor safety recovery while limiting interference with downstream task learning. Extensive experiments demonstrate the effectiveness and robustness of TDSR in preserving safety alignment while maintaining downstream task performance, outperforming the evaluated baselines. Our mechanism experiments suggest that TDSR offers an alternative route to safety recovery by forming new activation patterns for harmful inputs during repair, rather than simply preserving those of the original aligned model.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, assist with writing parts of the code, and identify potentially relevant studies. The authors carefully checked all AI-assisted material and reviewed all cited studies in detail.

ETHICS STATEMENT

This work aims to preserve the safety alignment of large language models during harmful fine-tuning while maintaining downstream task performance. We use harmful examples and adaptive attacks to assess vulnerabilities and evaluate the proposed defense for defensive research purposes. This work is intended to help model developers build more effective safeguards for the responsible fine-tuning of large language models.

REPRODUCIBILITY STATEMENT

All datasets and pretrained models used in this work are publicly available. To facilitate reproduction of our method and experimental results, we provide our code in the supplementary materials.

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.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Röttger, Dan Jurafsky, Tatsunori Hashimoto, and James Y Zou. Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. In *International Conference on Learning Representations*, volume 2024, pp. 34196–34216, 2024.

Liang Chen, Xueting Han, Li Shen, Jing Bai, and Kam-Fai Wong. Vulnerability-aware alignment: mitigating uneven forgetting in harmful fine-tuning. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Peijian Gu, Quan Wang, and Zhendong Mao. Improve safety training of large language models with safety-critical singular vectors localization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4941–4954, 2025.

Zihan Guan, Mengxuan Hu, Ronghang Zhu, Sheng Li, and Anil Vullikanti. Benign samples matter! fine-tuning on outlier benign samples severely breaks safety. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

William W Hager. Updating the inverse of a matrix. *SIAM review*, 31(2):221–239, 1989.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. *Advances in neural information processing systems*, 37:8093–8131, 2024.

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.

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

Zixuan Hu, Li Shen, Zhenyi Wang, Yongxian Wei, and Dacheng Tao. Adaptive defense against harmful fine-tuning for large language models via bayesian data scheduler. *Advances in Neural Information Processing Systems*, 38:52131–52174, 2026.

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim F 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, 2024a.

Tiansheng Huang, Sihao Hu, and Ling Liu. Vaccine: Perturbation-aware alignment for large language models against harmful fine-tuning attack. *Advances in Neural Information Processing Systems*, 37:74058–74088, 2024b.

Tiansheng Huang, Gautam Bhattacharya, Pratik Joshi, Joshua Kimball, and Ling Liu. Antidote: post-fine-tuning safety alignment for large language models against harmful fine-tuning attack. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025a.

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Tekin, and Ling Liu. Booster: Tackling harmful fine-tuning for large language models via attenuating harmful perturbation. In *International Conference on Learning Representations*, volume 2025, pp. 67202–67226, 2025b.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36:24678–24704, 2023.

Richard D Lange, Devin Kwok, Jordan Kyle Matelsky, Xinyue Wang, David Rolnick, and Konrad Kording. Deep networks as paths on the manifold of neural representations. In *Proceedings of 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML)*, volume 221 of *Proceedings of Machine Learning Research*, pp. 102–133. PMLR, 2023. URL https://proceedings.mlr.press/v221/lange23a.html.

Simon Lermen and Charlie Rogers-Smith. LoRA fine-tuning efficiently undoes safety training in llama 2-chat 70b. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024. URL https://openreview.net/forum?id=Y52UbVhglu.

Changyue Li, Jiaming He, Youliang Yuan, Jialin Wu, Boxi Yu, Zhicong Huang, and Pinjia He. Trace: Trajectory-based safety patch learning for llm post-training realignment. *arXiv preprint arXiv:2607.16242*, 2026.

Mingjie Li, Wai Man Si, Michael Backes, Yang Zhang, and Yisen Wang. Salora: Safety-alignment preserved low-rank adaptation. In *International Conference on Learning Representations*, volume 2025, pp. 90827–90843, 2025.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 5 2023.

Guozhi Liu, Weiwei Lin, Qi Mu, Tiansheng Huang, Ruichao Mo, Yuren Tao, and Li Shen. Targeted vaccine: Safety alignment for large language models against harmful fine-tuning via layer-wise perturbation. *IEEE Transactions on Information Forensics and Security*, 2025.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

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.

Guoxin Lu, Letian Sha, Qing Wang, Peijie Sun, ZHOU HAO, Hua Dai, and Fu Xiao. Safety anchor: Defending harmful fine-tuning via geometric bottlenecks. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=BDR6RK9TmA.

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. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Mistral AI. Mistral-7B-Instruct-v0.3. Hugging Face model card, 2024. URL https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3.

Quoc Nguyen, Trung Le, Jing Wu, Anh Bui, and Mehrtash Harandi. Antibody: Strengthening defense against harmful fine-tuning for large language models via attenuating harmful gradient influence. In *International Conference on Learning Representations*, volume 2026, pp. 121713–121741, 2026.

OpenAI. GPT-5.6 Sol. OpenAI API model documentation, 2026. URL https://developers.openai.com/api/docs/models/gpt-5.6-sol. Accessed September 26, 2026.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. 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. 39643–39666. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/park24c.html.

ShengYun Peng, Pin-Yu Chen, Jianfeng Chi, Seongmin Lee, and Duen Horng Chau. Shape it up! restoring llm safety during finetuning. *Advances in Neural Information Processing Systems*, 38: 132053–132089, 2026.

Gabriel Jacob Perin, Runjin Chen, Xuxi Chen, Nina S. T. Hirata, Zhangyang Wang, and Junyuan Hong. Lox: Low-rank extrapolation robustifies LLM safety against fine-tuning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=ASS5YD4hL4.

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! In *International Conference on Learning Representations*, volume 2024, pp. 30988–31043, 2024.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In *International Conference on Learning Representations*, volume 2025, pp. 54911–54941, 2025.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Domenic Rosati, Jan Wehner, Kai Williams, Łukasz Bartoszcze, David Atanasov, Robie Gonzales, Subhabrata Majumdar, Carsten Maple, Hassan Sajjad, and Frank Rudzicz. Representation noising: A defence mechanism against harmful finetuning. *Advances in Neural Information Processing Systems*, 37:12636–12676, 2024.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5377–5400, 2024.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=9Vrb9D0WI4.

Han Shen, Pin-Yu Chen, Payel Das, and Tianyi Chen. SEAL: Safety-enhanced aligned LLM fine-tuning via bilevel data selection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=VHguhvcoM5.

Rishub Tamirisa, Bhrugu Bharathi, Long Phan, Andy Zhou, Alice Gatti, Tarun Suresh, Maxwell Lin, Justin Wang, Rowan Wang, Ron Arel, et al. Tamper-resistant safeguards for open-weight llms. In *International Conference on Learning Representations*, volume 2025, pp. 101802–101829, 2025.

Jiayu Tang, Guowei Peng, Qiuhao Xie, Yuning Yang, Xiurui Xie, and Guisong Liu. Oasis: Mitigating harmful fine-tuning attacks on llms via orthogonal and adaptive safety alignment strategy. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28407–28421, 2026.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Haozhong Wang, Zhuo Li, Yibo Yang, He Zhao, Hongyuan Zha, and Dandan Guo. Safeguarding llm fine-tuning via push-pull distributional alignment. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23624–23646, 2026a.

Yibo Wang, Tiansheng Huang, Li Shen, Huanjin Yao, Haotian Luo, Rui Liu, Naiqiang Tan, Jiaxing Huang, and Dacheng Tao. Panacea: Mitigating harmful fine-tuning for large language models via post-fine-tuning perturbation. *Advances in Neural Information Processing Systems*, 38:169951–169985, 2026b.

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. Assessing the brittleness of safety alignment via pruning and low-rank modifications. 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. 52588–52610. PMLR, 21–27 Jul 2024a. URL https://proceedings.mlr.press/v235/wei24f.html.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: empowering code generation with oss-instruct. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024b.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, et al. Sorry-bench: Systematically evaluating large language

model safety refusal. In *International Conference on Learning Representations*, volume 2025, pp. 59937–59973, 2025.

Shuo Yang, Qihui Zhang, Yuyang Liu, Yue Huang, Xiaojun Jia, Kun-Peng Ning, Jia-Yu Yao, Jigang Wang, Dai Hailiang, Yibing Song, et al. Asft: Anchoring safety during llm fine-tuning within narrow safety basin. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34322–34330, 2026.

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.

Biao Yi, Tiansheng Huang, Baolei Zhang, Tong Li, Lihai Nie, Zheli Liu, and Li Shen. Ctrap: Embedding collapse trap to safeguard large language models from harmful fine-tuning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10013–10030, 2026.

Xin Yi, Shunfan Zheng, Linlin Wang, Xiaoling Wang, and Liang He. A safety realignment framework via subspace-oriented model fusion for large language models. *Knowledge-Based Systems*, 306:112701, 2024.

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.

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori B Hashimoto, and Daniel Kang. Removing rlhf protections in gpt-4 via fine-tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 681–687, 2024.

Bingjie Zhang, Yibo Yang, Ren Zhe, Dandan Guo, Jindong Gu, Philip Torr, and Bernard Ghanem. A guardrail for safety preservation: When safety-sensitive subspace meets harmful-resistant null-space. In *International Conference on Learning Representations*, volume 2026, pp. 134322–134347, 2026.

Minjun Zhu, Linyi Yang, Yifan Wei, Ningyu Zhang, and Yue Zhang. Locking down the finetuned llms safety. *arXiv preprint arXiv:2410.10343*, 2024.

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. *arXiv preprint arXiv:2307.15043*, 2023b.

A METHOD PSEUDOCODE

Algorithm 1: TDSR

Input: Aligned parameters $\boldsymbol{\theta}_0$; fine-tuning data $\mathbb{D}_{\text{ft}}$; reference sets $\mathbb{D}_s, \mathbb{D}_t$; optimizer Opt; fine-tuning steps T; repair schedule $\mathbb{T}_{\text{repair}} \subseteq \{1, \dots, T\}$; repair step size $\eta_s > 0$; regularization $\rho > 0$; candidate count K.

Output: Adapted parameters $\boldsymbol{\theta}_{\text{ft}}$.

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_0$, $n_t \leftarrow |\mathbb{D}_t|$

for $t = 1, \dots, T$ **do**

  ▷ Downstream fine-tuning

  $\mathbb{B}_{\text{ft}} \leftarrow \text{NextBatch}(\mathbb{D}_{\text{ft}})$

  $\boldsymbol{\theta} \leftarrow \text{Opt}\Big(\boldsymbol{\theta}, \frac{1}{|\mathbb{B}_{\text{ft}}|} \sum_{(x,y)\in\mathbb{B}_{\text{ft}}} \nabla_{\boldsymbol{\theta}} \ell(x, y; \boldsymbol{\theta})\Big)$

  if $t \in \mathbb{T}_{\text{repair}}$ **then**

    ▷ Compute and cache task information for this repair

    $\boldsymbol{t}_j \leftarrow \nabla_{\boldsymbol{\theta}} \ell(x_j^t, y_j^t; \boldsymbol{\theta})$ for each $(x_j^t, y_j^t) \in \mathbb{D}_t$

    $s_t \leftarrow \text{median}_j \, \|\boldsymbol{t}_j\|_2, \quad \boldsymbol{Y} \leftarrow [\boldsymbol{t}_1, \dots, \boldsymbol{t}_{n_t}]^\top / (s_t \sqrt{n_t})$

    $\boldsymbol{L} \leftarrow \text{Cholesky}(\boldsymbol{I}_{n_t} + \rho^{-1} \boldsymbol{Y}\boldsymbol{Y}^\top)$, with $\boldsymbol{L}$ lower triangular

    foreach *mini-batch* $\mathbb{B}_s^{(r)}$ *from* $\mathbb{D}_s$ **do**

      ▷ Refresh safety gradients at the current parameters

      $n_s \leftarrow |\mathbb{B}_s^{(r)}|$

      $\boldsymbol{g}_i \leftarrow \nabla_{\boldsymbol{\theta}} \ell(x_i, y_i^{\text{safe}}; \boldsymbol{\theta})$ for each $(x_i, y_i^{\text{safe}}) \in \mathbb{B}_s^{(r)}$

      $s_s \leftarrow \text{median}_i \, \|\boldsymbol{g}_i\|_2, \quad \boldsymbol{X} \leftarrow [\boldsymbol{g}_1, \dots, \boldsymbol{g}_{n_s}]^\top / (s_s \sqrt{n_s})$

      $\boldsymbol{d} \leftarrow -\frac{1}{n_s} \sum_{i=1}^{n_s} \boldsymbol{g}_i$

      if $\boldsymbol{d} = \boldsymbol{0}$ **then continue**

      ▷ Compute reduced eigenpairs using the cached factor

      Solve $\boldsymbol{L}\boldsymbol{U} = \boldsymbol{Y}\boldsymbol{X}^\top$ for $\boldsymbol{U}$, then $\boldsymbol{L}^\top \boldsymbol{V} = \boldsymbol{U}$ for $\boldsymbol{V}$

      $\boldsymbol{W} \leftarrow \rho^{-1}\boldsymbol{X}^\top - \rho^{-2}\boldsymbol{Y}^\top \boldsymbol{V}, \quad \boldsymbol{Q} \leftarrow \boldsymbol{X}\boldsymbol{W}$

      Compute eigenpairs $\boldsymbol{Q}\boldsymbol{a}_k = \lambda_k \boldsymbol{a}_k$, ordered as $\lambda_1 \geq \cdots \geq \lambda_{n_s}$

      $q \leftarrow \min\big(K, \big|\{k : \lambda_k > 0\}\big|\big)$

      if $q = 0$ **then continue**

      for $k = 1, \dots, q$ **do**

        $\boldsymbol{v}_k \leftarrow \boldsymbol{W}\boldsymbol{a}_k$

        $\boldsymbol{u}_k \leftarrow \boldsymbol{v}_k / \|\boldsymbol{v}_k\|_2$

        if $\boldsymbol{d}^\top \boldsymbol{u}_k < 0$ **then** $\boldsymbol{u}_k \leftarrow -\boldsymbol{u}_k$

        $c_k \leftarrow \boldsymbol{d}^\top \boldsymbol{u}_k / \|\boldsymbol{d}\|_2$

      end

      ▷ Select a direction and apply the projected gradient

      $k^\star \leftarrow \arg\max_{1 \leq k \leq q} c_k$ (break ties by larger λ_k), $\quad \hat{\boldsymbol{v}} \leftarrow \boldsymbol{u}_{k^\star}$

      if $\boldsymbol{d}^\top \hat{\boldsymbol{v}} > 0$ **then** $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta_s (\boldsymbol{d}^\top \hat{\boldsymbol{v}}) \hat{\boldsymbol{v}}$

    end

  end

end

return $\boldsymbol{\theta}_{\text{ft}} \leftarrow \boldsymbol{\theta}$

B REGULARIZATION OF TASK SENSITIVITY

Adding $\rho \boldsymbol{I}$ to $\boldsymbol{A}_t$ makes the sensitivity ratio finite when reference gradients leave directions with zero empirical task sensitivity. It also controls how these directions are ranked during candidate search. The analysis below uses the normalized matrices in Equations 3 and 4, with positive normalization scales.

Degeneracy of the Empirical Task Sensitivity. The small task reference set yields a matrix $\boldsymbol{A}_t$ whose rank is limited by the number of reference gradients. With $n_t = |\mathbb{D}_t|$ and the gradient factor $\boldsymbol{Y} \in \mathbb{R}^{n_t \times m}$ defined in Equation 7,

$$\boldsymbol{A}_t = \boldsymbol{Y}^\top \boldsymbol{Y}, \qquad \text{rank}(\boldsymbol{A}_t) \leq n_t. \tag{13}$$

Thus, when $n_t < m$, $\boldsymbol{A}_t$ is singular. For a direction $\boldsymbol{v}$ in its null space,

$$\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v} = \frac{1}{n_t s_t^2} \sum_{j=1}^{n_t} (\boldsymbol{t}_j^\top \boldsymbol{v})^2 = 0 \quad \Longleftrightarrow \quad \boldsymbol{t}_j^\top \boldsymbol{v} = 0 \quad \text{for all } j. \tag{14}$$

This establishes zero first-order loss response only for the reference examples at $\boldsymbol{\theta}^{\mathrm{ref}}$. It does not establish zero sensitivity on unseen task examples or at later model parameters. The unregularized ratio $\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v} / (\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v})$ is undefined on such directions and therefore cannot provide finite scores for comparing them.

Finite Scores for Direction Comparison. The ridge term supplies a positive denominator for every nonzero direction while retaining safety sensitivity as the numerator. Since $\boldsymbol{A}_t$ is positive semidefinite and $\rho > 0$,

$$\boldsymbol{v}^\top \boldsymbol{B}_t \boldsymbol{v} = \boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v} + \rho \|\boldsymbol{v}\|_2^2 \geq \rho \|\boldsymbol{v}\|_2^2 > 0, \qquad \boldsymbol{v} \neq \boldsymbol{0}. \tag{15}$$

Consequently, $\boldsymbol{B}_t$ is positive definite and the objective in Equation 5 satisfies $0 \leq R(\boldsymbol{v}) \leq \lambda_{\max}(\boldsymbol{A}_s)/\rho$. The ratio is invariant to nonzero rescaling of $\boldsymbol{v}$ and attains a maximum over unit directions. For a unit direction in the null space of $\boldsymbol{A}_t$, it reduces to

$$R(\boldsymbol{v}) = \frac{\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v}}{\rho}, \qquad \|\boldsymbol{v}\|_2 = 1, \quad \boldsymbol{A}_t \boldsymbol{v} = \boldsymbol{0}. \tag{16}$$

These directions remain eligible for selection and are ordered by their empirical safety sensitivity. For example, two such directions with safety sensitivities a and $2a$, where $a > 0$, receive finite scores a/ρ and $2a/\rho$.

Effect of the Regularization Scale. The coefficient ρ determines how strongly the ratio distinguishes small empirical task sensitivities. For unit directions satisfying $\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v} \ll \rho$, the ratio is approximately $(\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v})/\rho$, so direction ranking depends primarily on safety sensitivity. When $\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v} \gg \rho$, it is approximately the unregularized ratio $\boldsymbol{v}^\top \boldsymbol{A}_s \boldsymbol{v} / (\boldsymbol{v}^\top \boldsymbol{A}_t \boldsymbol{v})$. Thus, the ridge term reduces the influence of very small denominators while preserving the relative task penalty when task sensitivity is large compared with ρ. This is a soft trade-off and does not impose a hard bound on task sensitivity.

Numerical Properties and Scope. Regularization also bounds the amplification introduced by applying $\boldsymbol{B}_t^{-1}$ in the reduced solver. If $\boldsymbol{A}_t \boldsymbol{u} = \mu \boldsymbol{u}$, then $\boldsymbol{B}_t \boldsymbol{u} = (\mu + \rho)\boldsymbol{u}$, so the eigenvalues are shifted by ρ while the eigenspaces are unchanged. In particular,

$$\|\boldsymbol{B}_t^{-1}\|_2 \leq \frac{1}{\rho}, \qquad \kappa_2(\boldsymbol{B}_t) = \frac{\lambda_{\max}(\boldsymbol{A}_t) + \rho}{\lambda_{\min}(\boldsymbol{A}_t) + \rho} \leq 1 + \frac{\lambda_{\max}(\boldsymbol{A}_t)}{\rho}. \tag{17}$$

The condition number can still be large when ρ is very small relative to the spectrum of $\boldsymbol{A}_t$. Although the task eigenspaces are preserved, changing ρ can change the generalized eigenvectors of $(\boldsymbol{A}_s, \boldsymbol{B}_t)$ and hence the candidate directions. These properties justify finite scoring and regularized computation, but do not guarantee task preservation on unseen data. Likewise, high safety sensitivity measures the magnitude of a local loss response, so Section 4.3 separately selects and orients updates to decrease the current safety loss to first order.

C Direction Search and Repair Update

C.1 Exact Reduced-Space Direction Search

This appendix proves Proposition 1 and explains how to compute the reduced problem without constructing matrices at the full parameter dimension. We use the gradient factors $\boldsymbol{X}$ and $\boldsymbol{Y}$ from Equation 7, with $\boldsymbol{A}_s = \boldsymbol{X}^\top \boldsymbol{X}$ and $\boldsymbol{B}_t = \rho \boldsymbol{I} + \boldsymbol{Y}^\top \boldsymbol{Y}$ for $\rho > 0$.

Proof of Proposition 1. Since $\boldsymbol{B}_t$ is positive definite, its inverse exists and $\boldsymbol{Q} = \boldsymbol{X} \boldsymbol{B}_t^{-1} \boldsymbol{X}^\top$ is symmetric positive semidefinite. Suppose $\boldsymbol{Q}\boldsymbol{a} = \lambda \boldsymbol{a}$ with $\boldsymbol{a} \neq \boldsymbol{0}$ and $\lambda > 0$. The recovered vector $\boldsymbol{v} = \boldsymbol{B}_t^{-1} \boldsymbol{X}^\top \boldsymbol{a}$ is nonzero, since $\boldsymbol{v} = \boldsymbol{0}$ would imply $\boldsymbol{Q}\boldsymbol{a} = \boldsymbol{0}$. Moreover,

$$\boldsymbol{A}_s \boldsymbol{v} = \boldsymbol{X}^\top \boldsymbol{X} \boldsymbol{B}_t^{-1} \boldsymbol{X}^\top \boldsymbol{a} = \boldsymbol{X}^\top \boldsymbol{Q} \boldsymbol{a} = \lambda \boldsymbol{X}^\top \boldsymbol{a} = \lambda \boldsymbol{B}_t \boldsymbol{v}. \tag{18}$$

Left-multiplying $\boldsymbol{A}_s\boldsymbol{v} = \lambda \boldsymbol{B}_t\boldsymbol{v}$ by $\boldsymbol{v}^\top$ gives

$$\boldsymbol{v}^\top \boldsymbol{A}_s\boldsymbol{v} = \lambda \boldsymbol{v}^\top \boldsymbol{B}_t\boldsymbol{v}.$$

Since $\boldsymbol{B}_t$ is positive definite and $\boldsymbol{v} \neq \boldsymbol{0}$, the denominator $\boldsymbol{v}^\top \boldsymbol{B}_t\boldsymbol{v}$ is strictly positive. Dividing both sides by this quantity yields

$$R(\boldsymbol{v}) = \frac{\boldsymbol{v}^\top \boldsymbol{A}_s\boldsymbol{v}}{\boldsymbol{v}^\top \boldsymbol{B}_t\boldsymbol{v}} = \frac{\lambda \boldsymbol{v}^\top \boldsymbol{B}_t\boldsymbol{v}}{\boldsymbol{v}^\top \boldsymbol{B}_t\boldsymbol{v}} = \lambda.$$

Conversely, let $\boldsymbol{v} \neq \boldsymbol{0}$ satisfy $\boldsymbol{A}_s\boldsymbol{v} = \lambda \boldsymbol{B}_t\boldsymbol{v}$ with $\lambda > 0$. Define

$$\boldsymbol{a} = \frac{\boldsymbol{X}\boldsymbol{v}}{\lambda}, \qquad \boldsymbol{v} = \boldsymbol{B}_t^{-1}\boldsymbol{X}^\top \boldsymbol{a}. \tag{19}$$

Here $\boldsymbol{a} \neq \boldsymbol{0}$, because $\boldsymbol{X}\boldsymbol{v} = \boldsymbol{0}$ would imply $\lambda \boldsymbol{B}_t\boldsymbol{v} = \boldsymbol{0}$, contradicting $\boldsymbol{v} \neq \boldsymbol{0}$. It follows that

$$\boldsymbol{Q}\boldsymbol{a} = \boldsymbol{X}\boldsymbol{B}_t^{-1}\boldsymbol{X}^\top \boldsymbol{a} = \boldsymbol{X}\boldsymbol{v} = \lambda \boldsymbol{a}. \tag{20}$$

For each $\lambda > 0$, these two maps are inverses on the corresponding eigenspaces, so their dimensions agree. Both problems are diagonalizable: $\boldsymbol{Q}$ is symmetric, and $\boldsymbol{B}_t^{-1}\boldsymbol{A}_s$ is similar to the symmetric matrix $\boldsymbol{B}_t^{-1/2}\boldsymbol{A}_s\boldsymbol{B}_t^{-1/2}$. Thus their positive eigenvalues also agree in algebraic multiplicity, proving the proposition.

Consequently, solving the reduced problem preserves the candidate directions associated with the largest positive sensitivity ratios, up to eigenvector scaling and the choice of basis within repeated eigenspaces. When $\boldsymbol{X} \neq \boldsymbol{0}$, recovering a direction from a largest eigenvalue of $\boldsymbol{Q}$ gives a maximizer of R. If $\boldsymbol{X} = \boldsymbol{0}$, then $\boldsymbol{A}_s = \boldsymbol{0}$ and $R(\boldsymbol{v}) = 0$ for every nonzero $\boldsymbol{v}$; no positive candidate exists, and the repair step is skipped.

Computation with Task Reference Gradients. The factorization $\boldsymbol{B}_t = \rho \boldsymbol{I} + \boldsymbol{Y}^\top \boldsymbol{Y}$ allows products with its inverse to be evaluated using an $n_t \times n_t$ system. For any $\boldsymbol{z} \in \mathbb{R}^m$, the Woodbury identity (Hager, 1989) gives

$$\boldsymbol{B}_t^{-1}\boldsymbol{z} = \frac{\boldsymbol{z}}{\rho} - \frac{1}{\rho^2}\boldsymbol{Y}^\top \left(\boldsymbol{I}_{n_t} + \frac{1}{\rho}\boldsymbol{Y}\boldsymbol{Y}^\top\right)^{-1} \boldsymbol{Y}\boldsymbol{z}. \tag{21}$$

In implementation, we evaluate the expression in parentheses through a Cholesky solve. The coefficient matrix remains fixed within each repair because $\boldsymbol{Y}$ does, so its Cholesky factorization is computed once and reused for subsequent safety mini-batches. For each mini-batch, applying Equation 21 to the columns of $\boldsymbol{X}^\top$ yields $\boldsymbol{B}_t^{-1}\boldsymbol{X}^\top$, from which we form $\boldsymbol{Q}$ and recover the candidate directions. The eigendecomposition therefore has dimension n_s, and the linear system has dimension n_t. The gradient factors and recovered directions still depend on the parameter dimension m, but no $m \times m$ matrix is constructed or inverted.

C.2 Descent Property of the Repair Update

The projected update in Section 4.3 is a descent direction for the current mini-batch safety loss whenever it is nonzero. At the current parameters $\boldsymbol{\theta}$, let $\boldsymbol{d} = -\nabla_{\boldsymbol{\theta}}\mathcal{L}_s(\boldsymbol{\theta})$ and $\hat{\boldsymbol{v}}$ be the selected unit direction. The vector $\boldsymbol{d}$ uses the original sample gradients without the sensitivity normalization by s_s. For $\eta_s > 0$, Equation 12 gives

$$\left(\nabla_{\boldsymbol{\theta}}\mathcal{L}_s\right)^\top \Delta\boldsymbol{\theta} = -\eta_s(\boldsymbol{d}^\top \hat{\boldsymbol{v}})^2 \leq 0. \tag{22}$$

If $\Delta\boldsymbol{\theta} \neq \boldsymbol{0}$, then $\boldsymbol{d}^\top \hat{\boldsymbol{v}} \neq 0$, and the inequality is strict. Differentiability of $\mathcal{L}_s$ therefore implies that this update decreases the current mini-batch loss for a sufficiently small positive η_s. This is a local property of the safety loss; task preservation is encouraged by the sensitivity criterion rather than guaranteed by this descent argument.

In implementation, ties in candidate alignment are resolved in favor of the larger sensitivity ratio $R(\boldsymbol{v}_k)$. The repair step skips a mini-batch if no positive candidate exists, $\boldsymbol{d} = \boldsymbol{0}$, or the selected projection is zero, as specified in Algorithm 1.

D EXPERIMENTAL DETAILS

D.1 MODELS AND DATA CONSTRUCTION

Fine-tuning Splits. Unless otherwise specified, harmful examples are randomly sampled from BeaverTails (Ji et al., 2023) and mixed with benign task examples at a harmful ratio of $p = 0.2$. The mathematical reasoning set contains 800 benign GSM8K examples and 200 harmful examples, while the code generation set contains 800 Python examples from Magicoder-OSS-Instruct-75K and 200 harmful examples. The instruction-following set contains 560 benign examples and 140 harmful examples. For code generation, 64 additional Magicoder Python examples are held out as task reference data. Where possible, the same benign and harmful fine-tuning splits are used across the four models.

Reference Data. The defender maintains 200 safety reference examples and 64 task reference examples for each downstream task, disjoint from the fine-tuning and evaluation data. The safety reference set $\mathbb{D}_s$, denoted Repair200, contains harmful prompts randomly sampled from BeaverTails (Ji et al., 2023) and paired with model-specific safe target responses. The task reference set $\mathbb{D}_t$ contains benign examples reserved from each downstream task dataset and held out from fine-tuning and evaluation. Harmful fine-tuning prompts are checked against Repair200 and Beaver700, and overlapping prompts are replaced.

D.2 EVALUATION PROTOCOLS

Mathematical Reasoning. GSM8K accuracy is evaluated on all 1,319 test examples. The prompt requests a final answer after `####`. The evaluator extracts that answer, falling back to the last numerical answer when the marker is absent.

Instruction Following. On the 105 held-out AlpacaEval prompts, GPT-5.6-Sol (OpenAI, 2026) blindly compares each evaluated model's responses with the text-davinci-003 reference responses provided by AlpacaEval (Li et al., 2023). The win rate is $100(W + 0.5T)/105$, where W and T denote wins and ties, respectively.

Code Generation. We measure pass@1 on HumanEval+ using EvalPlus (Liu et al., 2023) with 164 test samples. Prompts request a self-contained Python script in a Markdown code block. All utility metrics are reported as percentages.

Safety. Beaver700 contains 700 held-out harmful prompts. The Beaver-Dam classifier labels a generated response as harmful when its harmfulness score is greater than 0.5. Harmful Score (HS) is the percentage of responses labeled harmful.

D.3 TRAINING AND PERIODIC REPAIR

Downstream Training. Fine-tuning uses LoRA for 20 epochs in BF16 precision. Following the LoRA baseline setup in Zhang et al. (2026), we use rank-8 adapters on `q_proj` and `v_proj`. We set $\alpha = 16$. The micro-batch size is 1 with eight gradient accumulation steps. The learning rate starts at 10^{-5} and follows a cosine schedule to zero. The gradient norm is clipped at 1.0.

Repair Schedule. TDSR performs two repair events per epoch, for 40 events over downstream fine-tuning. Each event processes all 200 safety reference examples in 50 mini-batches of four examples from distinct safety categories. For code generation, each epoch contains 125 optimizer updates, with repair events after local updates 63 and 125. Task reference gradients are computed once per event, while safety gradients are recomputed for each mini-batch. The task and safety gradient collections are separately scaled by their median gradient norms, preserving relative gradient magnitudes within each collection.

Repair Updates. Repair uses the negative log-likelihood of the safe target responses as its loss. We set the candidate count to $K = 4$, the ridge coefficient to $\rho = 10^{-3}$, and the repair step size to $\eta_s = 2 \times 10^{-3}$.

D.4 Robustness Evaluation Protocols

Training Budgets. Using Qwen2.5-7B-Instruct and Magicoder at $p = 0.2$, we evaluate all methods after 10, 20, 30, 40, and 50 fine-tuning epochs. The 20-epoch results reuse the main-evaluation runs.

Harmful Ratios and Evaluation. The experiments at $p \in \{0, 0.1, 0.2, 0.3\}$ use independent 20-epoch runs. FA is pass@1 on the 164 HumanEval+ problems, and HS is the percentage of the 700 Beaver responses computed by a Beaver-Dam classifier.

D.5 Adaptive Attack Protocols

Attacker Access and Proxy Checkpoints. The attacks target the fine-tuning data with knowledge of the defense and its actual 64 task reference examples. For each model, attack construction uses 20 checkpoints from the corresponding TDSR control run with unmodified harmful data, one saved after the final safety repair of each training epoch. The attacks do not modify the safety or task reference sets or defense hyperparameters, and do not differentiate through the full training and repair procedure. Each attack run starts independently from the same base model and initial LoRA parameters as its control.

Gradient Objective. At a given control checkpoint, let $\mathbf{G}_t \in \mathbb{R}^{64\times m}$ contain the task reference loss gradients as rows, and let $\mathbf{g}_h \in \mathbb{R}^m$ be the negative log-likelihood gradient of the target response in a harmful training example h, both computed with respect to the trainable LoRA parameters. We compute the checkpoint-specific score

$$\widehat{\mathbf{g}}_h = \frac{\mathbf{g}_h}{\|\mathbf{g}_h\|_2}, \qquad S(\mathbf{g}_h) = \frac{\|\mathbf{G}_t \widehat{\mathbf{g}}_h\|_2^2}{\|\mathbf{G}_t\|_F^2}. \tag{23}$$

This score measures the task-gradient second moment along the normalized harmful gradient. It is unchanged by reversing that gradient and does not imply alignment with task descent or improvement in task performance. Let $S_e(h)$ denote this score for example h at the checkpoint from epoch e. Both attacks use the mean across the 20 checkpoints,

$$\overline{S}(h) = \frac{1}{20}\sum_{e=1}^{20} S_e(h). \tag{24}$$

Gradient Selection. The candidate pool combines the original 200 harmful examples with additional examples from the BeaverTails training partition. Candidates are ranked by $\overline{S}(h)$, and the highest-scoring candidates are selected to form 200 harmful examples in total. Selected examples replace the original harmful positions, leaving the 800 benign examples and the benign/harmful position sequence unchanged.

Gradient Suffix. A shared 16-token suffix is appended only to the 200 harmful training inputs, with their target responses unchanged. Discrete coordinate search proposes single-token replacements from printable ASCII tokens that round-trip through the tokenizer, excluding special tokens. The search uses 96 proposals and 32 fixed development examples, screening candidates on eight examples before evaluating the full development set. Candidates must satisfy an average gradient cosine of at least 0.5 relative to the original gradients, a gradient norm ratio of at least 0.1 for every development example, and a mean target-response loss increase of at most 0.5. All supervised target tokens must remain within the sequence-length limit. The remaining 168 harmful examples are used for diagnostics only, and final evaluation scores are not used to select the suffix. Suffix candidates are accepted based on improvements in the mean of $\overline{S}(h)$ over the development examples, subject to the constraints above; thus, suffix optimization also averages scores over all 20 checkpoints.

Training, Evaluation, and Scope. All four models use Magicoder with 800 benign and 200 harmful examples for 20 epochs, with the periodic repair schedule in Appendix D.3. Each attack is constructed independently for its target model. We evaluate HumanEval+ pass@1 and the harmful-response rate on the same task examples and 700 safety prompts without adding the attack suffix at evaluation. Comparisons in Section 5.4 use the standard TDSR results on Magicoder in Table 1 as the reference.

D.6 Representation Comparison Details

Prior work studies concept directions and representation control (Park et al., 2024; Zou et al., 2023a), refusal-mediating activation directions (Arditi et al., 2024), and safety-relevant parameter regions (Wei et al., 2024a).

Activation Extraction. We compare Qwen2.5-7B-Instruct models fine-tuned on Magicoder with the original aligned model (Base) using the same 700 harmful prompts from Beaver700. Each prompt is supplied without a chat template or an assistant prefix. For a prompt x, let $\mathbf{h}_l(x) \in \mathbb{R}^{3584}$ denote the activation at the last input token after Transformer block l, and let $\mathbf{h}_l^{\mathrm{Base}}(x)$ denote the corresponding Base activation. We use the residual output of each of the 28 blocks, before the final RMSNorm. Viewing successive layer representations as a trajectory (Lange et al., 2023), we compare both the layer activations and their increments

$$\Delta\mathbf{h}_l(x) = \mathbf{h}_l(x) - \mathbf{h}_{l-1}(x), \qquad l = 2, \ldots, 28. \tag{25}$$

These increments describe changes during the forward pass, not parameter updates during training.

Comparison Metrics. For each prompt and layer, we compute three distances from Base. Omitting x for readability, they are

$$d_{\mathrm{act},l} = \frac{\|\mathbf{h}_l - \mathbf{h}_l^{\mathrm{Base}}\|_2}{\|\mathbf{h}_l^{\mathrm{Base}}\|_2}, \tag{26}$$

$$d_{\mathrm{dir},l} = 1 - \frac{(\Delta\mathbf{h}_l)^\top \Delta\mathbf{h}_l^{\mathrm{Base}}}{\|\Delta\mathbf{h}_l\|_2 \, \|\Delta\mathbf{h}_l^{\mathrm{Base}}\|_2}, \tag{27}$$

$$d_{\mathrm{chg},l} = \frac{\|\Delta\mathbf{h}_l - \Delta\mathbf{h}_l^{\mathrm{Base}}\|_2}{\|\Delta\mathbf{h}_l^{\mathrm{Base}}\|_2}. \tag{28}$$

Figure 3(a) reports $d_{\mathrm{act},l}$, the relative distance between activations at the same layer. Panel (b) reports $d_{\mathrm{dir},l}$, which compares increment directions independently of their magnitudes. Panel (c) reports $d_{\mathrm{chg},l}$, which measures the relative difference between the full increment vectors and therefore reflects differences in both direction and magnitude. All evaluated activation and increment norms are nonzero.

Aggregation and Visualization. Each plotted point is the arithmetic mean of the corresponding per-prompt distance over the 700 inputs; activations are not averaged before distances are computed. Panel (a) includes layers 1–28, while panels (b,c) include the 27 transitions from layer 1 to layer 28, indexed by their destination layer. Metrics are computed in the original activation coordinates, without dimensionality reduction or rotation alignment. Larger distances indicate greater deviation from Base, rather than a ranking of safety performance. Panels (b,c) use symmetric logarithmic axes, with linear regions below 10^{-5} and 10^{-3}, respectively; this affects only visualization, not the reported distances.

E Parameter Sensitivity

We examine how repair frequency, candidate count K, repair step size η_s, and reference-set size affect the safety and task performance of TDSR. All experiments use Qwen2.5-7B-Instruct fine-tuned on Magicoder for 20 epochs at $p = 0.2$, with HS measured on Beaver700 and FA on HumanEval+. Unless varied, repair occurs every half epoch with $K = 4$, $\eta_s = 0.002$, and four safety examples per mini-batch.

E.1 Repair Frequency

More frequent repair consistently lowers HS while having a small effect on FA (Table 4). As the repair interval decreases from two epochs to a quarter epoch, HS falls from 5.71% to 4.14%, while FA ranges from 79.27% to 81.10% across all four intervals. These results support the effectiveness of TDSR in preserving downstream task performance even with frequent safety repair. Every repair processes the same 200 safety examples and refreshes the task gradients, so the four intervals entail 80, 40, 20, and 10 repairs, respectively. The comparison therefore reflects changes in both repair timing and total repair budget.

Table 4: Effect of repair frequency. HS and FA are percentages.

Repair interval (epochs)	HS ↓	FA ↑
1/4	4.14	79.27
1/2	4.57	81.10
1	5.14	80.49
2	5.71	79.88

Table 5: Effect of candidate count. $^{\dagger}K = 8$ uses eight safety examples per mini-batch instead of four. HS and FA are percentages.

K	HS ↓	FA ↑
2	6.00	80.49
3	4.29	79.27
4	4.57	81.10
$8^{\dagger}$	4.71	79.88

Table 6: Effect of repair step size. HS and FA are percentages.

η_s	HS ↓	FA ↑
0.0005	8.43	77.44
0.002	4.57	81.10
0.003	4.29	79.88
0.008	4.86	80.49

E.2 Candidate Count

Across $K \in \{2, 3, 4, 8\}$, TDSR maintains low HS of 4.29%–6.00% and high FA of 79.27%–81.10% (Table 5). The slightly higher HS of 6.00% at $K = 2$ suggests that retaining too few candidates may weaken safety recovery.

E.3 Repair Step Size

TDSR tolerates larger repair step sizes while preserving task performance, consistent with selecting directions that have little effect on task losses (Table 6). For $\eta_s \in \{0.002, 0.003, 0.008\}$, HS remains between 4.29% and 4.86%, with FA between 79.88% and 81.10%. By contrast, $\eta_s = 0.0005$ yields 8.43% HS and 77.44% FA, suggesting that an overly small step size may provide insufficient repair. We therefore recommend relatively larger repair step sizes within the tested range.

E.4 Reference Set Sizes

We evaluate the sensitivity of TDSR to the amount of safety and task reference data. The safety sweep uses $|\mathbb{D}_s| \in \{10, 50, 100, 200\}$ with $|\mathbb{D}_t| = 64$, and the task sweep uses $|\mathbb{D}_t| \in \{8, 16, 32, 64\}$ with $|\mathbb{D}_s| = 200$. Both use nested subsets of the default reference sets.

Controlled Repair Budget. Every configuration performs 40 repairs, each with 50 updates using four safety examples per update. Smaller safety sets reuse examples to maintain 200 sample uses per repair, so the safety sweep changes the number of distinct examples while keeping the repair-update budget fixed. Task gradients are refreshed once per repair, so their computation cost increases with task reference-set size.

Safety Reference Size. HS decreases as the safety reference set grows under the fixed repair-update budget, while FA shows no monotonic trend (Figure 4(a,b)). Increasing the set from 10 to

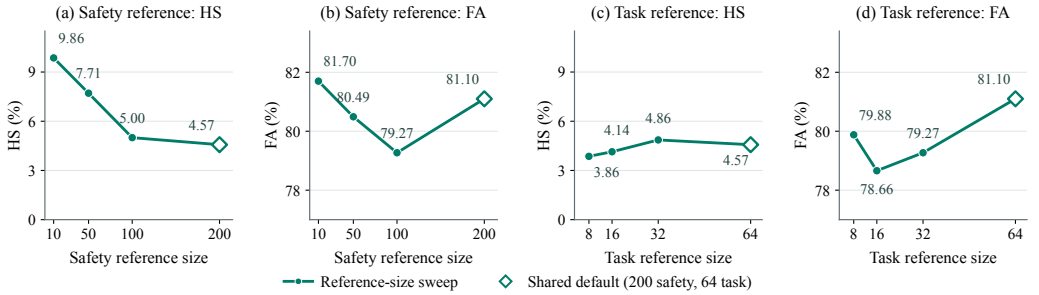

Figure 4: Sensitivity to reference-set size on Qwen2.5-7B-Instruct with Magicoder. Panels (a,b) show HS and FA as the safety reference set grows with 64 task examples, while panels (c,d) vary the task reference set with 200 safety examples. Hollow diamonds mark the shared default.

Table 7: Safety results (%) on additional harmful-request benchmarks. ASR uses the HarmBench classifier, CR is the SORRY-Bench compliance rate, and HS uses Beaver-Dam. Lower is better in every column. For each metric, the **best** and **second-best** results among fine-tuned models are marked. Base is an unadapted reference.

Method	AdvBench		HarmBench		HEx-PHI		SORRY-Bench	
	ASR	HS	ASR	HS	ASR	HS	CR	HS
Base	1.35	0.58	23.75	15.83	15.33	9.00	35.45	6.59
Vanilla SFT	66.73	68.85	85.00	64.17	82.67	67.67	92.05	52.73
LISA	0.77	**0.58**	32.50	19.58	18.67	7.33	40.91	14.32
SBR	0.77	1.15	**20.24**	20.00	**15.00**	10.00	**33.18**	11.59
Booster	**0.58**	**0.38**	32.50	**17.92**	15.67	9.67	38.64	**9.32**
OASIS	1.92	30.77	38.33	40.42	43.67	35.33	65.00	31.82
GuardSpace	1.15	0.96	34.17	20.83	17.33	**6.00**	33.41	10.23
TDSR	**0.19**	**0.38**	**21.25**	**15.00**	**9.33**	**4.67**	**27.73**	**7.73**

200 examples lowers HS from 9.86% to 4.57%, with FA ranging from 79.27% to 81.70% across the four sizes.

Task Reference Size. Across 8 to 64 task examples, HS ranges from 3.86% to 4.86% and FA from 78.66% to 81.10% (Figure 4(c,d)). The default 64-example set achieves the highest FA, while eight examples yield the lowest HS and 79.88% FA. These results suggest that even a small number of task reference examples can effectively guide repair while preserving task performance.

F Safety Generalization and Over-Refusal

We evaluate safety beyond Beaver700 and examine whether safety preservation leads to excessive refusal of benign requests. The experiments use Qwen2.5-7B-Instruct checkpoints after 20 epochs of Magicoder fine-tuning at $p = 0.2$.

F.1 Additional Safety Benchmarks

We evaluate direct harmful requests from AdvBench (Zou et al., 2023b), HarmBench (Mazeika et al., 2024), HEx-PHI (Qi et al., 2024), and SORRY-Bench (Xie et al., 2025). We report attack success rate (ASR) using the HarmBench-Llama-2-13b classifier on the first three datasets, with adapted templates for AdvBench and HEx-PHI. SORRY-Bench uses its dedicated Mistral-7B judge to measure compliance with harmful requests. We also report Beaver-Dam HS.

Table 8: Refusal and safety results (%) on XSTest. WildGuard measures over-refusal on 250 safe prompts and refusal on 200 harmful prompts. HS is measured by Beaver-Dam on the harmful prompts. Highlighting follows Table 7.

Method	Safe over-refusal ↓	Harmful refusal ↑	Harmful Score ↓
Base	4.40	91.00	0.50
Vanilla SFT	**0.40**	18.50	67.00
LISA	**2.40**	85.00	7.00
SBR	4.80	**91.00**	**2.00**
Booster	9.20	**92.00**	3.50
OASIS	6.40	70.50	18.50
GuardSpace	**2.40**	81.50	4.00
TDSR	4.00	**92.00**	**2.50**

Safety Results. TDSR reduces harmful responses relative to vanilla SFT on all four additional benchmarks under both evaluation protocols (Table 7). Under the benchmark-specific judges, TDSR achieves the lowest rates on AdvBench, HEx-PHI, and SORRY-Bench. Relative to the strongest baseline on each dataset, it lowers ASR by 0.39 and 5.67 percentage points on the first two and compliance by 5.45 percentage points on the third. On HarmBench, its ASR is 21.25%, compared with SBR's 20.24%. Beaver-Dam likewise assigns TDSR the lowest HS among the compared fine-tuned models on all four datasets, including a tie on AdvBench. These results extend the observed safety benefits beyond Beaver700.

F.2 Over-Refusal on XSTest

Excessive safety training can increase refusals of benign requests (Bianchi et al., 2024). XSTest (Röttger et al., 2024) tests whether models distinguish benign requests with potentially sensitive wording from harmful requests. We use WildGuard (Han et al., 2024) to classify response refusal, reporting over-refusal on 250 safe prompts and refusal on 200 harmful prompts. HS is evaluated separately on the harmful prompts, since refusal and harmfulness are distinct properties.

Refusal Results. TDSR combines 4.00% over-refusal with 92.00% refusal of harmful requests on XSTest (Table 8). These compare with 4.40% and 91.00% for Base, respectively. Although vanilla SFT has lower over-refusal at 0.40%, it refuses only 18.50% of harmful requests and reaches 67.00% HS, compared with 2.50% for TDSR. The results show that the observed safety benefits of TDSR are compatible with a low rate of unnecessary refusal in this setting.

G Additional Robustness Evaluation on Mistral

We extend the robustness evaluation to Mistral-7B-Instruct-v0.3 fine-tuned on Magicoder, varying fine-tuning duration at a harmful-data ratio of $p = 0.2$ and varying this ratio from 0 to 0.3 at 20 epochs. Figure 5 reports HS on Beaver700 and FA on HumanEval+ for TDSR and the six baselines.

Fine-tuning Duration. TDSR maintains the lowest HS at every evaluated training budget, with HS ranging from 12.57% to 15.71% and FA from 31.10% to 32.93% (Figure 5(a,b)). Its FA is highest at 20 and 50 epochs, and tied for the highest at 30 and 40 epochs. At 50 epochs, TDSR lowers HS from 36.43% to 15.14% and raises FA from 30.49% to 32.32% relative to the strongest baseline for each metric, improvements of 21.29 and 1.83 percentage points, respectively.

Harmful-Data Ratio. TDSR achieves the lowest HS and highest FA at all three nonzero harmful-data ratios, including a tie in FA at $p = 0.1$ (Figure 5(c,d)). At $p = 0.3$, TDSR lowers HS from 29.43% to 14.71% and raises FA from 30.49% to 33.54% relative to the strongest baseline for each metric. Across all four ratios, TDSR keeps HS between 12.29% and 14.71% and FA between 31.71% and 33.54%.

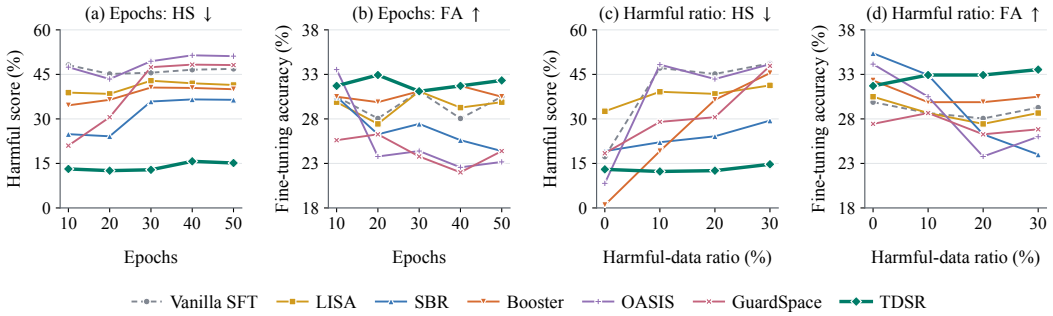

Figure 5: Robustness evaluation on Mistral-7B-Instruct-v0.3 with Magicoder. Panels (a,b) report HS and FA as fine-tuning duration varies at $p = 0.2$, while panels (c,d) vary the harmful-data ratio at 20 epochs.

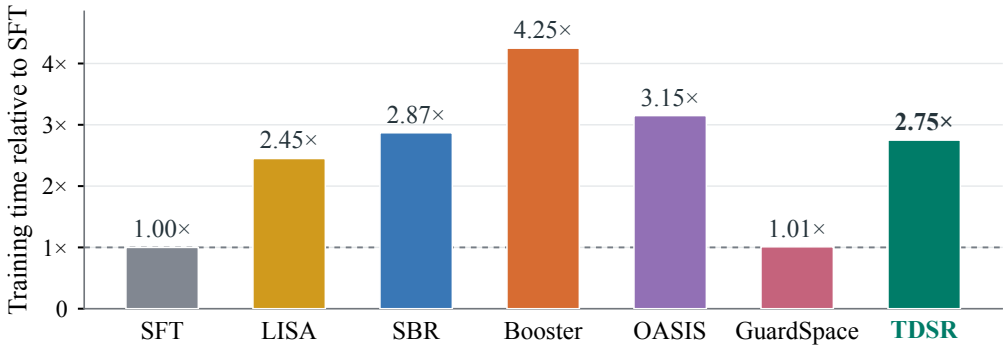

Figure 6: Total training time on Qwen2.5-7B-Instruct with GSM8K, normalized to SFT (1.00×). Measurements include all training stages. Lower is better.

H Training Efficiency

We evaluate training efficiency on Qwen2.5-7B-Instruct with GSM8K, measuring the total training time of each method relative to vanilla SFT. The measurements include the complete training pipeline, including any preliminary alignment and safety repair. As shown in Figure 6, TDSR takes 2.75× the SFT training time, close to LISA (2.45×) and lower than SBR (2.87×), OASIS (3.15×), and Booster (4.25×). These results place TDSR below three of the five defense baselines in training time, supporting its practical use despite the additional direction search and repair.

I Evaluation on Qwen3.8-27B

We extend the evaluation to Qwen3.8-27B, comparing TDSR with vanilla SFT, SBR, and GuardSpace on GSM8K, AlpacaEval, and Magicoder after 20 epochs at $p = 0.2$. Table 9 reports HS and FA. TDSR achieves the lowest HS on all three tasks and the highest FA on AlpacaEval and Magicoder, while retaining 91.81% accuracy on GSM8K.

Table 9: Results (%) on Qwen3.8-27B across three downstream tasks after 20 epochs at $p = 0.2$. For each dataset and metric, the best result is **bold with a green background**, while the second best is **bold with a gray background**.

Dataset	Method	Harmful Score ↓	Fine-tuning Accuracy ↑
GSM8K	Vanilla SFT	50.14	**93.18**
	SBR	6.00	**91.96**
	GuardSpace	**5.43**	88.10
	TDSR	**3.71**	91.81
AlpacaEval	Vanilla SFT	39.71	55.24
	SBR	**3.71**	**60.48**
	GuardSpace	4.00	55.71
	TDSR	**2.43**	**65.71**
Magicoder	Vanilla SFT	52.71	83.54
	SBR	**5.43**	**86.59**
	GuardSpace	7.14	82.32
	TDSR	**4.57**	**88.41**