Separate the Wheat from the Chaff: A Post-Hoc Approach to Safety Re-Alignment for Fine-Tuned Language Models

Anonymous ACL submission

Abstract

Although large language models (LLMs) achieve effective safety alignment at the time of release, they still face various safety challenges. A key issue is that fine-tuning often compromises the safety alignment of LLMs. To address this issue, we propose a method named **IRR** (**I**dentify, **R**emove, and **R**ecalibrate for Safety Realignment) that performs safety realignment for LLMs. The core of IRR is to identify and remove unsafe delta parameters from the fine-tuned models, while recalibrating the retained ones. We evaluate the effectiveness of IRR across various datasets, including both full fine-tuning and LoRA methods. Our results demonstrate that IRR significantly enhances the safety performance of fine-tuned models on safety benchmarks, such as harmful queries and jailbreak attacks, while maintaining their performance on downstream tasks. The source code is available at: https://anonymous.4open.science/r/IRR-BD4F.

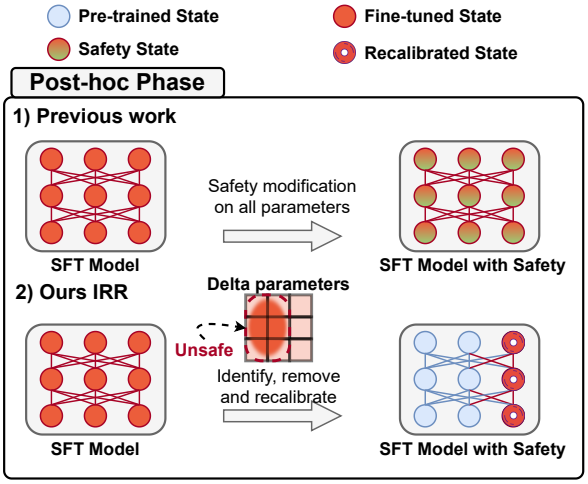


Figure 1: The illustration presents post-hoc approaches for safety realignment. Our method, IRR, first identifies and removes unsafe delta parameters, then recalibrates the remaining ones. The IRR approach enhances safety while maintaining performance.

1 Introduction

In recent years, large language models (LLMs) have been widely used due to their significant success in various tasks (Qin et al., 2023; Zhao et al., 2023b). A common paradigm for LLMs is "release and fine-tuning." Before release, developers conduct safety alignment to achieve a safety-aligned model (Ouyang et al., 2022). After release, these LLMs are made available through fine-tuning APIs or open-source platforms, enabling users to further fine-tune them for specific downstream tasks.

In the "release and fine-tuning" paradigm, LLMs acquire delta parameters through fine-tuning, which enhance their performance on downstream tasks. However, this process often compromises the safety mechanisms established during safety alignment, reducing their value as reliable AI services. Specifically, training data that mixes harmful data with benign data, or even consists entirely of benign data, can significantly compromise the safety alignment of LLMs (Bhardwaj and Poria, 2023; Wan et al., 2023; Zhan et al., 2024; Qi et al., 2023). Given the widespread adoption of the "release and fine-tuning" paradigm and its associated risks, a key objective is to ensure the safety realignment of fine-tuned models while maintaining their performance on downstream tasks.

To achieve this objective, a straightforward method is to directly modify the parameters of fine-tuned models. A notable example is RESTA (Bhardwaj et al., 2024), which improves model safety by incorporating a safety vector into all parameters (Figure 1). However, current methods overlook the safety relevance and task-specific relevance of model parameters (Frankle and Carbin, 2018; Panigrahi et al., 2023; Wei et al., 2024), as well as the trade-offs between them. As a result, applying the same safety modifications to all parameters without distinction may unintentionally harm those critical for downstream task performance. In-

spired by recent work on removing redundant delta parameters from fine-tuned models (Panigrahi et al., 2023; Yu et al., 2024), we propose separating unsafe delta parameters from fine-tuned models. As shown in Figure 1, by removing unsafe delta parameters and recalibrating the retained parameters, we can achieve safety realignment while maintaining performance on downstream tasks.

Based on these insights, we propose **IRR** (**I**dentify, **R**emove, and **R**ecalibrate for Safety Realignment), a simple yet effective post-hoc method for the safety realignment of fine-tuned models. Specifically, the IRR method involves three steps: (1) We use safety vectors to represent the parameter changes that move the model from an unsafe state to a safety-aligned state. Next, to identify unsafe delta parameters, our identification strategy combines safety importance and safety interference which result from sign disagreement with the safety vector. (2) We remove the unsafe delta parameters from the fine-tuned models. The remaining delta parameters are retained. (3) Unsafe delta parameters can compromise model safety but may be important for downstream task performance. Therefore, removing unsafe delta parameters can lead to performance degradation. To mitigate this, we recalibrate the retained delta parameters using precise weight compensation based on the inverse of the Hessian matrix.

We conducted extensive experiments to evaluate IRR under full fine-tuning and LoRA fine-tuning across various datasets. Compared to the baselines, IRR significantly improves model safety while preserving downstream task performance, achieving a Pareto improvement.

The main contributions of this work are summarized as follows:

- We propose IRR, a novel safety realignment method that improve safety through three steps: identify, remove, and recalibrate.
- IRR introduces a novel perspective on safety realignment, showing that combining safety interference and safety importance scores can effectively separate unsafe delta parameters from fine-tuned models to enhance safety.
- Extensive experiments across various datasets, fine-tuning methods, and models show that IRR effectively restores safety while preserving downstream task performance, achieving Pareto improvements.

2 Related Work

LLMs Safety The safety of LLMs aims to mitigate potential safety risks arising from misuse or malicious use. Recent studies have identified vulnerabilities in the safety alignment of LLMs. Yang et al. (2023); Bhardwaj and Poria (2023); Zhan et al. (2024) demonstrated that even fine-tuning on small amounts of harmful data can significantly impact the safety alignment of LLMs. Qi et al. (2023) used more practical datasets, such as identity shift data and benign data like Alpaca, to undermine the safety alignment of LLMs.

To address the safety compromises introduced by fine-tuning, current methods focus on three main phases: (1) Pre-processing phase, where Zhao et al. (2023a) uses catastrophic forgetting to filter harmful data; (2) Fine-tuning phase, where Huang et al. (2024) limits parameter updates to reduce safety loss; (3) Post-hoc phase, where Bhardwaj et al. (2024) employs a merging safety vector approach to enhance model safety, and Zhao et al. (2024) introduces a patch to the safety vector to mitigate over-safety issues. Additionally, Hsu et al. (2024) proposes the Safe LoRA method, and Yi et al. (2024) utilizes sub-network search techniques to train a safety sub-network within the fine-tuned model. Inference-time safety has also gained attention, with Hazra et al. (2024) removing harmful vectors and adjusting the latent space, and Xu et al. (2024) improving content safety by increasing the probability of safe tokens during decoding.

We focus on the post-hoc phase that does not require additional fine-tuning, thereby reducing computational costs while allowing flexible trade-offs between safety and downstream task performance.

Supervised Fine-Tuning and Delta Parameters. Supervised fine-tuning (SFT) is a widely used method to enhance the performance of pre-trained LLMs on specific downstream tasks. This process involves changing the model parameters to improve task performance, with these alterations referred to as delta parameters. Recent studies have highlighted redundancy in these delta parameters in fine-tuned models. Panigrahi et al. (2023) addressed this issue by employing sub-network search to selectively prune delta parameters, retaining only a minimal subset necessary to achieve performance comparable to standard SFT. Similarly, Yu et al. (2024) introduced the DARE method, which involves randomly dropping a certain proportion of delta parameters and rescaling the remaining ones.

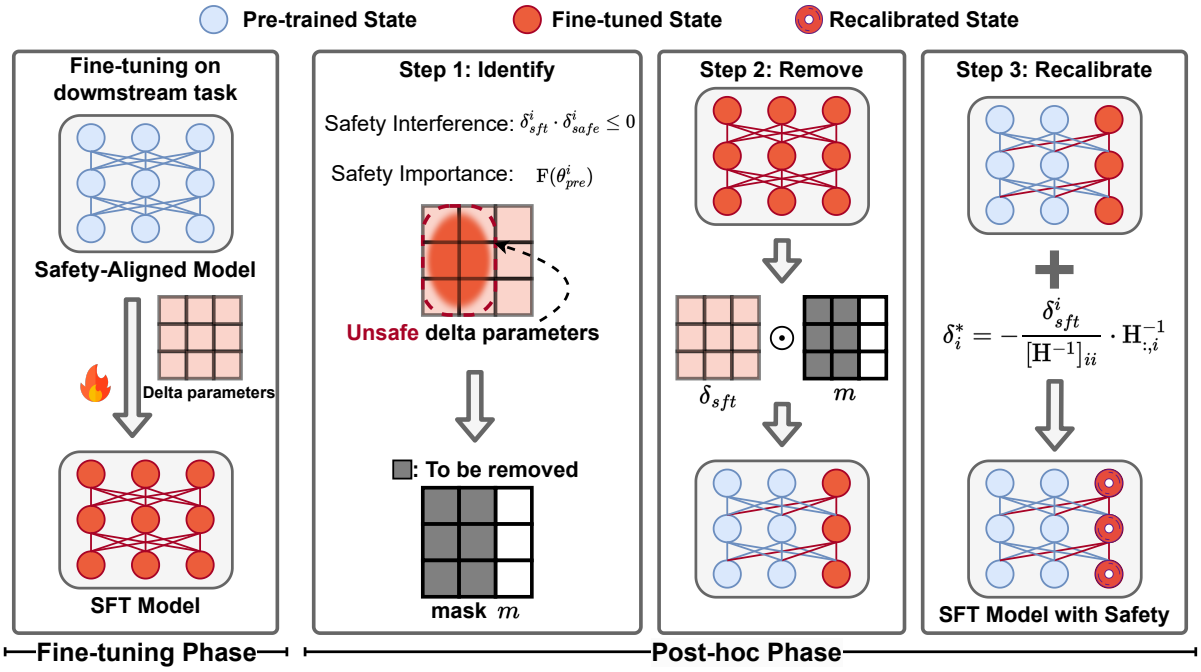


Figure 2: During fine-tuning phase, safety-aligned models acquire delta parameters that enhance downstream task performance, but these parameters may compromise model safety. In the post-hoc phase, IRR carefully identifies and removes unsafe delta parameters. It then computes compensatory values and adds them to the retained parameters, effectively restoring safety while preserving the model performance on downstream tasks.

Zhu et al. (2024) proposed the use of significance and sensitivity metrics to identify critical delta parameters. Our method also focuses on delta parameters, but emphasizes their role in balancing safety and downstream performance, rather than focusing solely on downstream tasks.

Model Pruning Technique As neural network models grow in size, model pruning techniques have been widely adopted to reduce computational costs (Cheng et al., 2017; Liang et al., 2021). The goal of model pruning is to remove unnecessary parameters while maintaining model performance (Zhu and Gupta, 2017). The key to successful pruning lies in evaluating the importance of parameters and removing those with low importance scores to compress the model. For example, Liu et al. (2021) used the Fisher matrix to compute parameter importance and removed those with low scores. Frantar and Alistarh (2023) introduced SparseGPT, forming importance scores by solving a layer-wise reconstruction problem, while Sun et al. (2024) proposed the Wanda score, which assesses parameter importance using joint weight/activation metrics. Although our approach shares the concept of parameter removal with model pruning, our focus is on removing certain delta parameters rather than removing entire parameters.

3 Approach

We propose IRR, a novel method for safety realignment of fine-tuned models, which effectively restores safety while maintaining downstream task performance. The overall framework of IRR is illustrated in Figure 2, and consists of the following three steps: (1) **Identify the Unsafe Delta Parameters**. This step identifies delta parameters that interfere with important parameters for safety alignment and marks these interfering delta parameters as unsafe. (2) **Remove the Unsafe Delta Parameters**. These unsafe delta parameters are removed and reverted to their original safe pre-trained states, improving the safety of fine-tuned models. (3) **Recalibrate the Retained Parameters**. Since some unsafe delta parameters may significantly affect downstream task performance, removing them could degrade performance. To mitigate this, we compute compensatory values and add them to the retained parameters.

3.1 Identify the Unsafe Delta Parameters

In this step, we propose two strategies: **Safety Interference** and **Safety Importance** to identify the unsafe delta parameters. These strategies help separate unsafe delta parameters from fine-tuned models, thereby restoring safety.

Safety Interference To identify unsafe delta parameters, it is crucial to clarify the direction of safe parameter updates. Therefore, we define a safety vector δ_{safe}, which represents the parameter differences when moving from the unaligned model to the safety-aligned model:

$$\delta_{\text{safe}} = \theta_{\text{align}} - \theta_{\text{unalign}} \tag{1}$$

Inspired by the concept of interference (Yadav et al., 2024), we hypothesize that if a delta parameter δ^i has a sign disagreement with the safety vector δ^i_{safe}, it causes safety interference that compromises model safety. Therefore, it is essential to identify delta parameters in δ_{sft} that exhibit sign disagreement with δ_{safe}, forming a candidate set $\mathcal{U}$ of safety interference delta parameters, as defined by the following formula:

$$\mathcal{U} = \{\delta^i_{\text{sft}} \in \delta_{\text{sft}} \mid \delta^i_{\text{sft}} \cdot \delta^i_{\text{safe}} \leq 0, \forall i\} \tag{2}$$

Safety Importance After identifying the candidate set of safety interference delta parameters $\mathcal{U}$, the next step is to determine which of these parameters poses a threat to model safety. Building on previous work (Liu et al., 2021; Matena and Raffel, 2022), we introduce the Fisher matrix (Fisher, 1922; Amari, 1996) as a safety importance score to evaluate the significance of each parameter relative to the safety alignment of the original model.

To simplify computation for LLMs, we approximate the Fisher matrix by averaging the gradients of N samples to estimate its diagonal (Kirkpatrick et al., 2017). Our estimation is as follows:

$$\hat{F}_\theta = \frac{1}{N} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{y \sim p_\theta(y|x_i)} (\nabla_\theta \log p_\theta(y|x_i))^2 \tag{3}$$

where $x_1, \ldots, x_N$ represent harmful queries, and the expectation over y indicates a safe refusal response that refuses harmful queries. Notably, the Fisher matrix is computed on the original model before fine-tuning and can be reused in the post-hoc phase without repeated computation.

Parameters with high safety importance scores are critical for the safety alignment of the original model. If delta parameters that cause safety interference exist on high-importance parameters, they may compromise the safety alignment of the model. Such delta parameters should be considered unsafe.

To identify the unsafe delta parameters, we extract the parameters in the top $\rho\%$ based on their safety importance scores from the set $\mathcal{U}$ and apply a mask to designate these delta parameters as unsafe. Additionally, s' denotes the score of the parameter at the $\rho\%$ position within $\mathcal{U}$. The method for determining the final mask m is defined as follows:

$$m_i = \begin{cases} 1, & \text{if } \delta^i_{\text{sft}} \in \mathcal{U} \text{ and } s_i \geq s' \\ 0, & \text{otherwise} \end{cases}. \tag{4}$$

Finally, we identify the delta parameters with a mask value of 1 as unsafe.

3.2 Remove the Unsafe Delta Parameters

In this step, we remove the identified unsafe delta parameters while retaining the remaining ones. We define the removing process. For the delta parameters δ_{sft}, we introduce a mask $m \in \{0,1\}^{|\theta|}$ to indicate which delta parameters are unsafe and will be removed. Meanwhile, θ_{pre} denotes the parameters of the pre-trained safety-aligned model. The parameters of the model are computed as follows:

$$\widetilde{\theta}_{\text{sft}} = (1 - m) \odot \delta_{\text{sft}} + \theta_{\text{pre}}. \tag{5}$$

3.3 Recalibrate the Retained Parameters

Removing unsafe delta parameters improves model safety but may degrade downstream performance, as some unsafe parameters are critical for tasks. To address this, we add compensatory values δ^*_{sft} to the retained delta parameters $\widetilde{\delta}_{sft}$, identified by the mask m. During this step, these retained parameters are recalibrated to maintain task performance.

$$\hat{\theta}_{\text{sft}} = \{\widetilde{\theta}^i_{\text{sft}} + \delta^*_i \mid m_i = 0, \widetilde{\theta}^i_{\text{sft}} \in \widetilde{\theta}_{\text{sft}}\} \tag{6}$$

Previous work on the Optimal Brain Surgeon (OBS) theory (LeCun et al., 1989; Hassibi et al., 1993; Zhu et al., 2024) analyzed the change in loss caused by parameter alterations and studied the minimal perturbation required for the remaining parameters to minimize the loss. Based on these theories, our method applies compensatory values δ^*_i to the retained delta parameters θ^i_{sft}, ensuring optimal performance on downstream tasks. The compensatory values for the retained parameters are computed using the following formula:

$$\delta^*_i = -\frac{\theta^i_{\text{sft}} - \theta^i_{\text{pre}}}{[\mathrm{H}^{-1}]_{ii}} \cdot \mathrm{H}^{-1}_{:,i}, \tag{7}$$

Here, H^{-1} represents the inverse of the Hessian matrix, and $\mathrm{H}^{-1}_{:,m}$ denotes the m-th column of H^{-1}.

The identify, remove, and recalibrate steps are executed iteratively on the parameter matrix using a specified block size, continuing this process until the entire parameter matrix has been traversed.

4 Experimental Setup

We conducted experiments using both full fine-tuning and LoRA (Hu et al., 2022). The results for full fine-tuning are presented in the main text, while LoRA results are in Appendix B.

Model Our experiments are conducted on the widely used open-source model Llama-2-7b-chat (Touvron et al., 2023), which has been fine-tuned to follow instructions, align with human preferences, and ensure strong safety. Additionally, we perform LoRA fine-tuning experiments based on the Llama-3-8B-Instruct (Dubey et al., 2024). Supervised fine-tuning (SFT) is conducted using the LLaMA Factory [1] (Zheng et al., 2024), and the resulting models are referred to as domain-specific fine-tuned models.

Dataset For Llama-2, we utilized three datasets to obtain the SFT models: GSM8K (Cobbe et al., 2021) for Math, CodeAlpaca-20k [2] for Code, and Chinese Alpaca (Taori et al., 2023) for Chinese capability. Following the setting of Bhardwaj et al. (2024), we incorporated an additional 50K English instances into the Chinese Alpaca dataset to ensure the ability of the model to respond to English instructions. For Llama-3, we use the MathInstruct (Yue et al., 2024) dataset to obtain the SFT model.

Baselines IRR and IRR$_d$ refer to the application of the method on SFT models without and with DARE (Yu et al., 2024), respectively. We compared the IRR and IRR$_d$ method against several baselines:

- **SFT** involves fine-tuning on downstream task data using a language modeling objective.
- **DARE** (Yu et al., 2024) applies a drop-and-rescale operation on the delta parameters of the SFT model.
- **Safe LoRA** (Hsu et al., 2024) maps the delta parameter matrix of the fine-tuned model into the subspace of safe vectors, resulting in a more secure fine-tuned model.
- **SafeDecoding** (Xu et al., 2024) identifies and amplifies the probabilities of safe tokens in generated content while reducing the probabilities of unsafe tokens, thereby enhancing model safety.
- **RESTA** (Bhardwaj et al., 2024) improves the safety of fine-tuned model by incorporating safety vectors. Specifically, **RESTA** and **RESTA**$_d$ refer to methods that integrate safety vectors into SFT models without and with DARE (Yu et al., 2024), respectively.

Computing Safety Vectors and Fisher Matrix According to Bhardwaj et al. (2024), we define the safety vector δ_{safe} as the difference in parameters between the aligned and unaligned models. The unaligned model is fine-tuned using a harmful question-answer dataset. We extracted 1,000 labeled harmful question-answer pairs from the BeaverTails dataset (Ji et al., 2024) for training.

To compute the Fisher matrix, we relied on the same set of harmful questions but generated safe responses using the aligned model to create a safety dataset. This safety dataset serves as the calibration dataset for computing the Fisher matrix.

Evaluation Setup To comprehensively evaluate the safety and robustness of LLMs, we considered two evaluation setups: (1) Harmful query benchmark and (2) Jailbreak attacks.

For the harmful query benchmark, we utilized three datasets: 1) CATQA (Bhardwaj et al., 2024), a multilingual dataset encompassing English, Chinese, and Vietnamese; 2) HEx-PHI (Qi et al., 2023), which contains 330 harmful queries collected based on the usage policies of Meta and OpenAI; 3) Salad-Base (Li et al., 2024), covering 6 domains, 16 tasks, and 66 categories. We performed stratified sampling on 10% of the Salad-Base dataset and obtained a total of 2,100 harmful queries.

To assess robustness against jailbreak attacks, we used the Salad-Attack (Li et al., 2024) dataset, which simulates various attack attempts using methods from GPTFuzzer (Yu et al., 2023), TAP (Mehrotra et al., 2023), GCG (Zou et al., 2023), AutoDAN (Liu et al., 2024), and human-designed templates, all derived from the Salad-Base dataset.

We evaluated the downstream task performance of the SFT model using GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), and the Chinese version of MMMLU [3] (Hendrycks et al., 2021a). For Llama3, we conducted evaluations on various mathematical tasks, with detailed settings provided in the Appendix E.

Evaluation Metrics We utilize MD-Judge (Li et al., 2024), a content moderation model based

[1] https://github.com/hiyouga/LLaMA-Factory
[2] https://huggingface.co/datasets/sahil2801/CodeAlpaca-20k
[3] https://github.com/openai/simple-evals

Method	Safety Score					Jailbreak Safety Score							Math↑
	CATQA	HEx-PHI	Salad-Base	Avg	Δ↓	GPTFuzz	TAP	GCG	AutoDAN	Template	Avg	Δ↓	(GSM8K)
SFT	78.85	71.52	90.01	80.12	19.83	32.11	44.76	23.91	36.96	38.52	35.25	47.71	**43.06**
DARE	78.97	71.21	90.43	80.20	19.75	32.51	41.43	23.91	36.10	38.58	34.51	48.05	42.99
SafeDecoding	93.52	95.45	96.90	95.29	4.66	78.39	85.24	92.12	52.44	52.44	76.35	6.05	38.67
Safe LoRA	99.88	100.00	99.86	**99.91**	**0.04**	74.73	96.19	89.13	65.04	89.58	**82.93**	**0.03**	22.61
RESTA	99.33	99.09	99.34	99.26	0.69	54.51	76.19	89.13	86.82	78.24	76.98	5.98	41.93
RESTA$_d$	99.52	98.79	99.20	99.17	0.78	55.10	75.71	88.86	84.24	77.49	76.28	6.68	41.77
IRR	99.58	99.39	99.72	99.56	0.39	59.56	82.86	85.05	91.12	79.68	79.65	3.31	42.91
IRR$_d$	99.70	99.70	99.77	99.72	0.23	60.36	84.29	86.96	93.12	82.10	82.10	0.86	42.61

(a) The results of fine-tuning on GSM8K and performing safety realignment.

Method	Safety Score					Jailbreak Safety Score							Code↑
	CATQA	HEx-PHI	Salad-Base	Avg	Δ↓	GPTFuzz	TAP	GCG	AutoDAN	Template	Avg	Δ↓	(HumanEval)
SFT	64.48	62.42	83.07	69.99	29.96	24.98	40.00	17.39	25.79	29.04	27.44	55.92	**19.02**
DARE	64.18	63.03	83.26	70.16	29.79	26.16	42.38	19.02	25.50	28.91	28.40	54.56	18.66
SafeDecoding	92.73	86.06	96.72	91.83	8.12	37.66	65.71	36.68	28.37	47.83	43.25	39.71	17.62
Safe LoRA	99.94	99.70	99.91	**99.85**	**0.10**	77.60	96.67	60.87	80.80	80.82	79.35	3.61	11.89
RESTA	99.70	96.97	99.62	98.76	1.19	66.30	81.90	68.21	93.12	83.47	78.60	4.36	14.88
RESTA$_d$	99.58	96.67	99.67	98.64	1.31	66.80	82.86	69.57	91.69	83.63	78.91	4.05	15.61
IRR	99.27	97.88	99.72	98.96	0.99	71.46	85.24	67.93	91.69	89.28	81.12	1.84	18.96
IRR$_d$	99.58	98.48	99.81	99.29	0.66	75.12	87.62	71.20	93.12	90.59	**83.53**	**-0.57**	**19.02**

(b) The results of fine-tuning on CodeAlpaca-20k and performing safety realignment.

Method	Safety Score					Jailbreak Safety Score							Chinese↑
	CATQA	HEx-PHI	Salad-Base	Avg	Δ↓	GPTFuzz	TAP	GCG	AutoDAN	Template	Avg	Δ↓	(MMMLU)
SFT	89.09	85.76	95.31	90.05	0.90	66.20	46.19	34.78	64.47	69.42	56.21	26.75	36.85
DARE	88.79	85.45	95.22	89.82	10.13	66.20	48.57	34.24	65.33	69.88	56.84	26.12	36.78
SafeDecoding	96.61	91.21	97.70	95.17	4.78	81.27	66.19	62.77	76.22	77.79	72.85	10.11	23.29
Safe LoRA	99.88	100.00	99.86	**99.91**	**0.04**	74.73	96.19	89.13	65.04	89.58	82.93	0.03	22.61
RESTA	98.91	98.18	99.39	98.83	1.12	91.28	78.57	79.89	98.85	96.01	**88.92**	**-5.96**	33.03
RESTA$_d$	99.03	98.18	99.48	98.90	1.05	89.89	77.62	79.35	99.14	95.95	88.39	-5.43	32.40
IRR	98.91	98.18	99.39	98.83	1.12	91.58	78.10	62.23	99.43	95.23	85.31	-2.35	**37.08**
IRR$_d$	98.85	99.39	99.58	99.27	0.68	92.17	81.90	62.23	100.00	95.95	86.45	-3.49	36.82

(c) The results of fine-tuning on Alpaca Chinese and performing safety realignment.

Table 1: We evaluate safety using harmful benchmarks and jailbreak attacks. A higher **safety score** indicates better safety, while Δ represents the difference in safety score compared to the original model. Higher performance in downstream tasks reflects better capability. The best and second-best results are highlighted in **bold** and underlined.

on LLMs, to evaluate the harmfulness of question-answer pairs, including responses to harmful requests and jailbreak attacks. We report the safety of the model as **Safety score**, defined as the proportion of responses assessed as harmless by MD-Judge to all annotated responses. A higher score indicates a safer model.

5 Results and Discussions

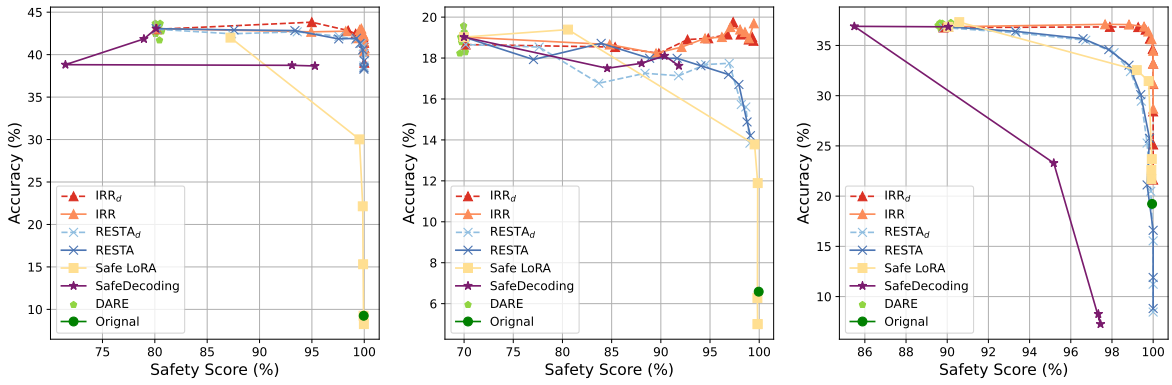
As shown in Table 1, all methods except DARE improved the safety of the SFT model. Although Safe LoRA enhanced model safety, it significantly reduced downstream task performance. For example, the accuracy on Math tasks dropped from 43.06 to 22.61. This indicates that projecting delta parameters into a safe subspace may disrupt parameters that are critical for downstream tasks.

The RESTA method also improved safety, with performance degradation varying across tasks. For example, the accuracy on Math tasks decreased by only 1.13, while accuracy on Code decreased by 4.14 and accuracy on Chinese decreased by 3.82. We also observed that the random dropout and scaling operations in the DARE method did not significantly improve safety, even when combined with RESTA or IRR, as seen in RESTA$_d$ and IRR$_d$.

In contrast, our IRR method maintains downstream task performance almost unchanged while enhancing safety compared to the SFT model.

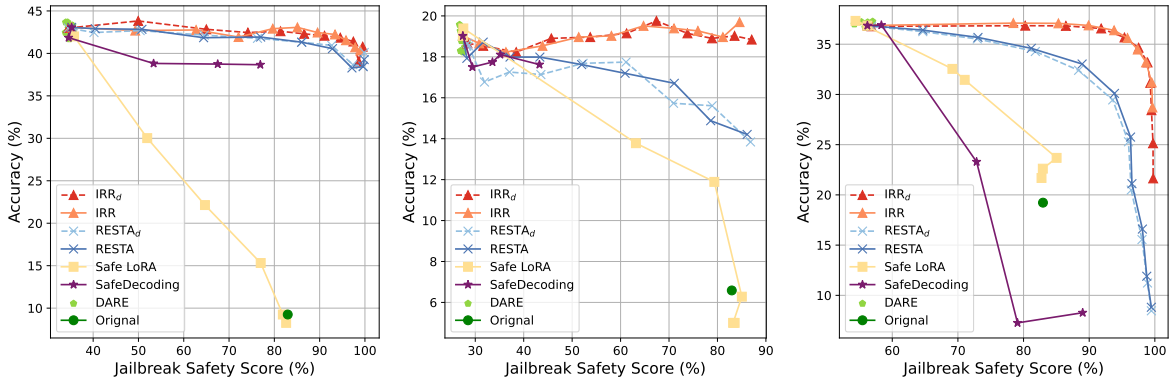
5.1 IRR Achieves Pareto Improvement

To investigate the trade-off between downstream task performance and safety during the safety improvement, we plotted the relationship between performance and safety (see Figure 3 and Figure 4). In the harmful benchmark and jailbreak attack, we observed that both RESTA and RESTA$_d$ maintained stable performance in the initial stages of safety improvements. However, as safety increased, their performance gradually declined, particularly for Code and Chinese tasks. In contrast, both IRR and IRR$_d$ consistently performed well at the safety frontier. Notably, even at safety levels close to those of the original model, IRR outperformed RESTA and



(a) Math (b) Code (c) Chinese

Figure 3: We show the trend of "downstream task performance vs. safety score" based on the **Harmful Benchmark**. Our method, IRR, outperforms baseline methods, maintaining downstream task performance as safety improves.



(a) Math (b) Code (c) Chinese

Figure 4: We show the trend of "downstream task performance vs. safety score" based on the **Jailbreak Attack**. Our method, IRR, outperforms baseline methods, maintaining downstream task performance as safety improves.

RESTA$_d$ in downstream task performance, especially in jailbreak attacks. Additionally, the LoRA experiment results detailed in Appendix B followed a similar trend as full fine-tuning, confirming the effectiveness of the IRR method.

5.2 Ablation Study

We conducted an ablation study on the IRR method and reported the "downstream task performance vs. safety" trade-off curves on jailbreak attacks in Figure 5. Additional ablation experiments can be found in the Appendix D.

Identifying the Unsafe Parameters. We ablated the identification step (IRR w/o ID) and replaced it with a random selection of delta parameters to be removed. As shown in Figure 5, skipping the identification step typically results in a significant drop in downstream task performance as a trade-off for improved safety. This highlights the critical importance of identifying unsafe delta parameters.

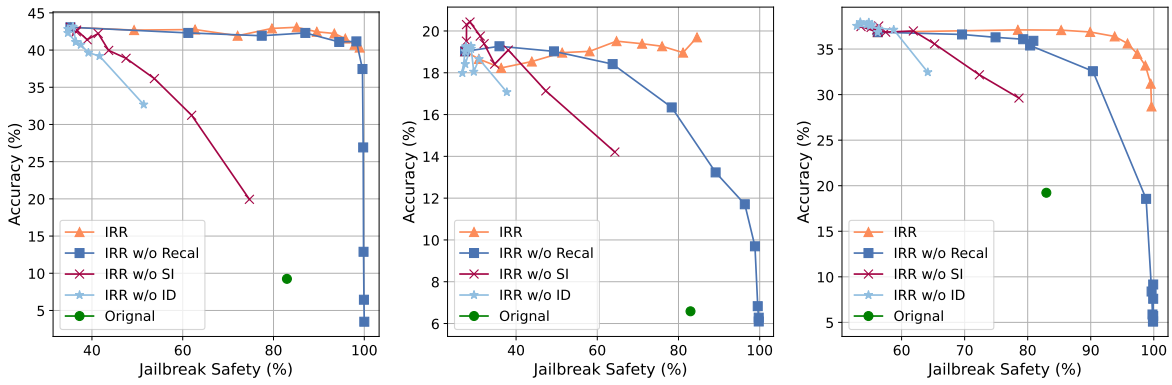
Safety Interference. IRR uses safety interference together with the safety importance score to identify unsafe delta parameters. We ablated the safety interference strategy in the identification step (IRR w/o SI). As shown in Figure 5, relying only on the safety importance score to identify unsafe delta parameters also leads to significant degradation in downstream task performance.

Recalibration. We ablated the recalibrate step (IRR w/o Recal). As shown in Figure 5, removing recalibration resulted in performance degradation, although the impact was relatively minor.

These results validate the effectiveness of the IRR method.

5.3 Cross-Language Safety Improvement

We assessed the safety of SFT models fine-tuned on mathematical datasets using the English, Chinese, and Vietnamese versions of the harmful benchmark CATQA (see Figure 6). Additional experiments



(a) Math (b) Code (c) Chinese

Figure 5: We present the results of the IRR ablation study using "downstream task performance vs. safety" curves.

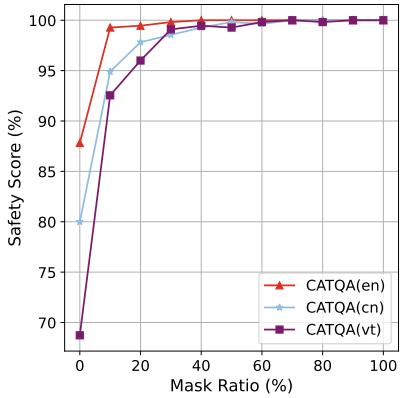


Figure 6: We conduct a safety evaluation on the English, Chinese, and Vietnamese versions of CATQA. We compare the safety changes of harmful queries across different mask ratios and languages.

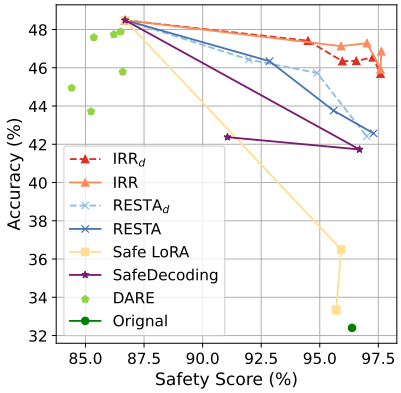


Figure 7: We fine-tune the Llama-3-8B-Instruct model using LoRA on the MathInstruct dataset. We then evaluate the impact of IRR and baseline methods on the safety and mathematical capabilities of the SFT model.

are provided in Appendix C.

We observed that SFT models achieve lower safety scores in Chinese and Vietnamese compared to English. As the IRR mask ratio increases, the safety scores for harmful queries across different languages gradually improve, eventually approaching the safety level of the original model. Additionally, as shown in Appendix C, different SFT models exhibit similar trends.

5.4 Efficacy of IRR Across Models

The IRR method is not restricted by any specific model architecture, allowing it to be applied across various models. To validate this claim, we conducted experiments by LoRA fine-tuning the LLama-3-8B-Instruct (Dubey et al., 2024). The experimental results on harmful benchmarks are shown in Figure 7. We evaluated the performance on several mathematical tasks and reported the average scores. The results demonstrate that for fine-tuned Llama3, the IRR method effectively improves model safety while maintaining downstream task performance, confirming its effectiveness. More detailed experimental results can be found in Appendix E.

6 Conclusion

In this paper, we introduce a safety realignment method IRR, which enhances model safety by identifying and removing unsafe delta parameters while recalibrating the remaining ones. IRR significantly improves the effectiveness of safety realignment. Evaluations on various harmful query benchmarks and jailbreak attacks indicate that IRR considerably reduces the risks of fine-tuned models. Among various fine-tuning methods, datasets, and models, IRR outperforms baseline methods by improving model safety while maintaining downstream task performance, achieving Pareto improvements.

7 Limitations

Our work explores the important issue of safety realignment in fine-tuned models. While our findings offer valuable insights, they also highlight several limitations and directions for future research.

Multimodal Models. Due to budget constraints, we did not conduct experiments on multimodal models. However, we believe that safety assessments for images, speech, and other modalities could reveal more interesting insights, which we plan to consider in future work.

Our IRR. Given the complex architecture of LLMs, our approach for obtaining safety vectors and evaluating the safety importance scores of parameters in the IRR method is relatively simple. Developing more robust and precise methods for these steps is necessary and should be a focus of future investigations.

Despite these limitations, we believe our work makes a new contribution to the field of safety alignment.

8 Ethical consideration

Ensuring ethical applications of artificial intelligence is critical. Our safety realignment method IRR enhances the safety of fine-tuned language models by reducing harmful content. The identification and removal operations effectively reduce harmful responses in fine-tuned models, while calibration ensures strong downstream task performance. Our framework demonstrates its effectiveness in improving the safety of fine-tuned models across different datasets. We advocate for ongoing collaboration among researchers, policymakers, and industry stakeholders to ensure that artificial intelligence development prioritizes human values, fairness, and safety. We remain committed to continuously evaluating and improving our approach to address ethical challenges.

9 Potential Risks

We now discuss the potential risks associated with our work. First, we highlight that the safety of fine-tuned models may be compromised, which could pose safety threats to users relying on these models for downstream tasks. We believe that improving safety will help the community benefit from advancements in secure large language models.

On the other hand, our proposed safety realignment method may lead users to mistakenly believe that the resulting models are completely safe, which may not be the case. We only demonstrate improvements in safety based on the evaluations presented in this paper. This also poses potential safety risks to users. We recommend exercising caution when deploying language models and always conducting safety checks.

References

Shun-ichi Amari. 1996. Neural learning in structured parameter spaces-natural riemannian gradient. *Advances in neural information processing systems*, 9.

Rishabh Bhardwaj, Do Duc Anh, and Soujanya Poria. 2024. Language models are homer simpson! safety re-alignment of fine-tuned language models through task arithmetic. *arXiv preprint arXiv:2402.11746*.

Rishabh Bhardwaj and Soujanya Poria. 2023. Language model unalignment: Parametric red-teaming to expose hidden harms and biases. *arXiv preprint arXiv:2310.14303*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. 2017. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*.

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

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

RA Fisher. 1922. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London Series A*, 222:309–368.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Babak Hassibi, David G Stork, and Gregory J Wolff. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE.

Rima Hazra, Sayan Layek, Somnath Banerjee, and Soujanya Poria. 2024. Safety arithmetic: A framework for test-time safety alignment of language models by steering parameters and activations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21759–21776, Miami, Florida, USA. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. 2024. Safe lora: the silver lining of reducing safety risks when fine-tuning large language models. *arXiv preprint arXiv:2405.16833*.

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

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, and Ling Liu. 2024. Lazy safety alignment for large language models against harmful fine-tuning. *arXiv preprint arXiv:2405.18641*.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2024. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Lijun Li, Bowen Dong, Ruohui Wang, Xuhao Hu, Wangmeng Zuo, Dahua Lin, Yu Qiao, and Jing Shao. 2024. SALAD-bench: A hierarchical and comprehensive safety benchmark for large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 3923–3954, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. 2021. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 461:370–403.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Liyang Liu, Shilong Zhang, Zhanghui Kuang, Aojun Zhou, Jing-Hao Xue, Xinjiang Wang, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. 2021. Group fisher pruning for practical network compression. In *International Conference on Machine Learning*, pages 7021–7032. PMLR.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2024. Autodan: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations*.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. 2023. Tree of attacks: Jailbreaking black-box llms automatically. *arXiv preprint arXiv:2312.02119*.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. 2022. NumGLUE: A suite of fundamental yet challenging mathematical reasoning tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3505–3523, Dublin, Ireland. Association for Computational Linguistics.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. 2023. Task-specific skill localization in fine-tuned language models. In *International Conference on Machine Learning*, pages 27011–27033. PMLR.

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

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1339–1384.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: an instruction-following llama model (2023). *URL https://github. com/tatsu-lab/stanford_alpaca*, 1(9).

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. 2023. Poisoning language models during instruction tuning. In *International Conference on Machine Learning*, pages 35413–35425. PMLR.

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. 2024. Assessing the brittleness of safety alignment via pruning and low-rank modifications. In *Forty-first International Conference on Machine Learning*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. 2024. SafeDecoding: Defending against jailbreak attacks via safety-aware decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5587–5605, Bangkok, Thailand. Association for Computational Linguistics.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2024. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36.

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

Xin Yi, Shunfan Zheng, Linlin Wang, Xiaoling Wang, and Liang He. 2024. A safety realignment framework via subspace-oriented model fusion for large language models. *arXiv preprint arXiv:2405.09055*.

Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. 2023. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. Mammoth: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori B Hashimoto, and Daniel Kang. 2024. 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)*, pages 681–687.

Jiachen Zhao, Zhun Deng, David Madras, James Zou, and Mengye Ren. 2023a. Learning and forgetting unsafe examples in large language models. *arXiv preprint arXiv:2312.12736*.

Weixiang Zhao, Yulin Hu, Zhuojun Li, Yang Deng, Yanyan Zhao, Bing Qin, and Tat-Seng Chua. 2024. Towards comprehensive and efficient post safety alignment of large language models via safety patching. *arXiv preprint arXiv:2405.13820*.

Weixiang Zhao, Yanyan Zhao, Xin Lu, Shilong Wang, Yanpeng Tong, and Bing Qin. 2023b. Is chatgpt equipped with emotional dialogue capabilities? *arXiv preprint arXiv:2304.09582*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2024. AGIEval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2299–2314, Mexico City, Mexico. Association for Computational Linguistics.

Didi Zhu, Zhongyi Sun, Zexi Li, Tao Shen, Ke Yan, Shouhong Ding, Kun Kuang, and Chao Wu. 2024. Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. *arXiv preprint arXiv:2402.12048*.

Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

A Experimental Details

Hyperparameter Settings For the Math, Code, and Chinese in Table 1, we set varying values for ρ, specifically 60%, 10%, and 80%, respectively. However, for each task, we can optimize the hyperparameters to achieve the best trade-off between downstream task performance and safety, as illustrated in Figure 3 and Figure 4. The calculation of compensatory values in the recalibration step requires downstream task data. For this purpose, we extracted 1,000 samples from the corresponding downstream task dataset. We employed greedy decoding as our generation strategy. All experiments were conducted on 4 $\times$ A100 80GB GPUs.

We conducted full fine-tuning experiments on Llama-2-7B-chat (Touvron et al., 2023), following the default configuration settings of Llama2. The initial learning rate was set to 2.0×10^{-5} and gradually decayed to zero using a cosine annealing schedule. The training batch size was set to 64. The number of epochs was set to 3, except for fine-tuning on the Alpaca Chinese dataset, for which only 1 epoch was used.

For experiments using Low-Rank Adaptation (LoRA) (Hu et al., 2022) to fine-tune Llama-2-7B-chat (Touvron et al., 2023), the query and value matrices in LoRA were adjusted with a rank of $r = 8$. We followed the default configuration settings of Llama2. The initial learning rate was set to 2.5×10^{-4} and gradually decayed to zero using a cosine annealing schedule. The training batch size was set to 64. The epoch was set to 5.

Llama-2 training and inference are conducted with the default system prompt.

B LoRA fine-tuning Experimental

We also conducted experiments on SFT models fine-tuned with LoRA (Hu et al., 2022) and evaluated their safety using the harmful benchmark and jailbreak attack. The specific experimental results are shown in Figure 8 and Figure 9. We found that for LoRA fine-tuning, the IRR method is also effective, achieving Pareto improvements in both safety enhancement and maintaining downstream task performance.

C Multilingual Safety

We assessed the safety of SFT models fine-tuned on the Code (CodeAlpaca-20k) and Chinese (Alpaca-Chinese) datasets using the English, Chinese, and Vietnamese versions of the harmful benchmark CATQA (see Figure 10). As shown in Figure 10, the safety scores across different languages improve as the masking ratio increases for both the Code and Chinese datasets.

D Ablation Experiments

We conducted an ablation study on the IRR method and presented the trade-off curve between downstream task performance and safety on the harmful benchmark in Figure 11.

E Details about Llama-3-8B-Instruct experiment

In the Llama3 experiments, we fine-tuned Meta-Llama-3-8B-Instruct (Dubey et al., 2024) using LoRA. The query and value matrices in LoRA were tuned with a rank of $r = 8$. The training batch size was set to 64, and the learning rate configured as 2.5×10^{-4}. Fine-tuning was performed on the MathInstruct dataset (Yue et al., 2024).

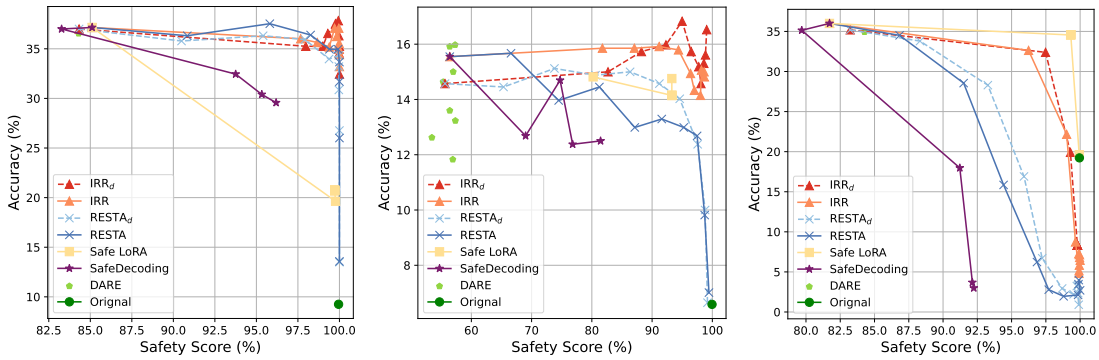
To evaluate the mathematical capabilities of the fine-tuned model, we conducted few-shot evaluations on GSM8K (Cobbe et al., 2021), Math (Hendrycks et al., 2021b), AQuA (Ling et al., 2017), simuleq (Koncel-Kedziorski et al., 2016), numglue (Mishra et al., 2022), MMLU STEM (Hendrycks et al., 2021a), and SAT math (Zhong et al., 2024) datasets. The evaluation was implemented using the math-evaluation-harness framework [4].

Meta-Llama-3-8B-Instruct does not include a default system prompt, so no system prompt is added during training or inference.

F Computational Complexity of IRR

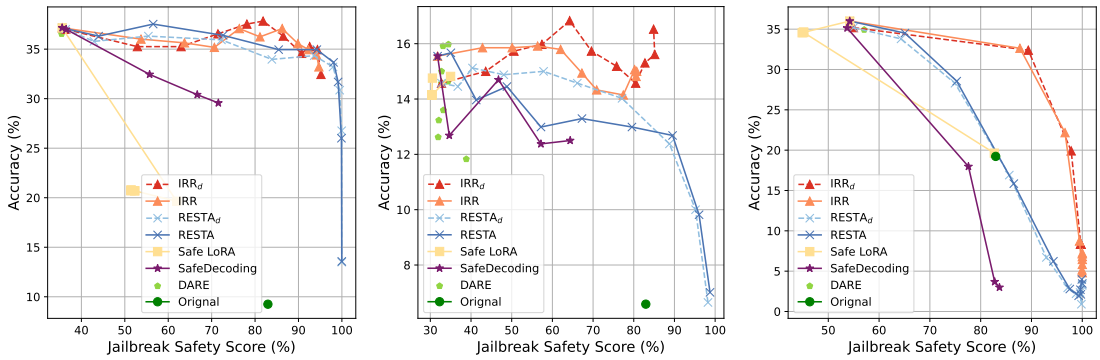
To implement IRR, we leverage a computationally efficient technique called SparseGPT (Frantar

[4] https://github.com/ZubinGou/math-evaluation-harness



(a) Math (b) Code (c) Chinese

Figure 8: LoRA experimental results. We present the trends of "task performance versus safety score" on the **harmful benchmark**. Our method, IRR, outperforms baseline methods by improving safety while maintaining strong downstream task performance.



(a) Math (b) Code (c) Chinese

Figure 9: LoRA experimental results. We present the trends of "task performance versus safety score" on the **Jailbreak attacks**. Our method, IRR, outperforms baseline methods by improving safety while maintaining strong downstream task performance.

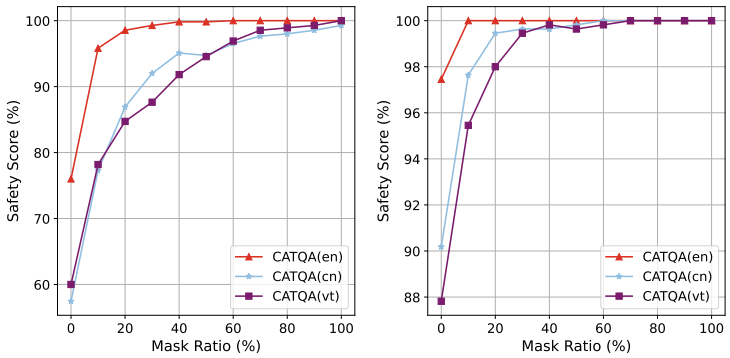
and Alistarh, 2023) to compute the inverse Hessian matrix, which is a critical component of the OBS computation. The computational complexity of calculating the inverse Hessian as described in SparseGPT can be divided into three main parts:

Initial Hessian Calculation. The time complexity for calculating the initial Hessian matrix is $O(nd_{\text{col}}^2)$, where n represents the number of input samples and d_{col} is the number of columns in the matrix.

Hessian Inversion Iteration. The iterative inversion of the Hessian matrix has a time complexity of $O(d_{\text{col}}^3)$, which remains manageable even for large models.

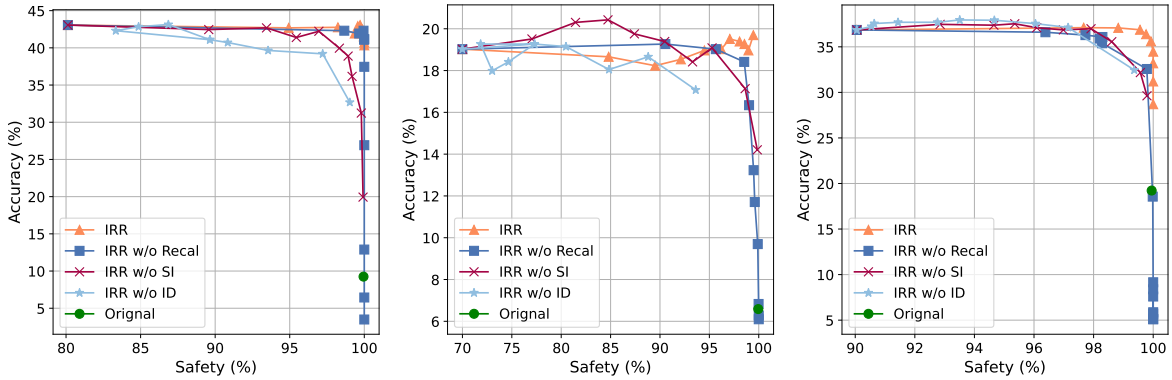
Reconstruction Process. The pruning or reconstruction process based on the inverse Hessian involves a complexity of $O(d_{\text{col}}^3 + d_{\text{row}}^2 d_{\text{col}})$, where d_{row} denotes the number of rows in the matrix. This ensures that the process is computationally feasible even for models with a large number of parameters.

In summary, considering the hidden dimension d_{hidden} of Transformer models, the overall computational complexity aligns with $O(d_{\text{hidden}}^3)$. This indicates a significant improvement in efficiency compared to exact reconstruction methods, demonstrating that our approach is computationally practical even for very large models.



(a) Code (b) Chinese

Figure 10: We conduct a safety evaluation on the English, Chinese, and Vietnamese versions of CATQA. We compare the safety changes of harmful queries across different mask ratios and languages.



(a) Math (b) Code (c) Chinese

Figure 11: We present the results of the IRR ablation study using "task performance vs. safety" curves on harmful benchmarks. Our method effectively identifies unsafe delta parameters and, combined with the calibration step, successfully preserves downstream task performance.

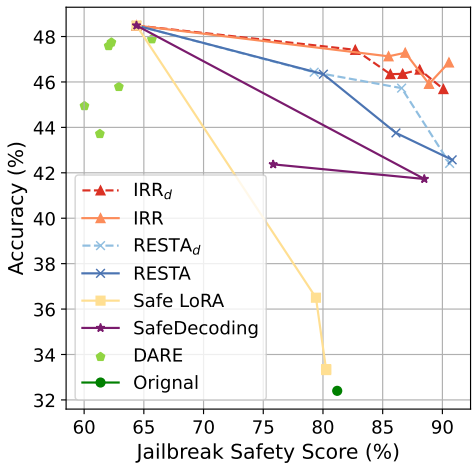


Figure 12: We fine-tune the Llama-3-8B-Instruct model using LoRA on the MathInstruct dataset. We then evaluate the impact of IRR and baseline methods on the safety and mathematical capabilities of the SFT model.