Improve Safety Training of Large Language Models with Safety-Critical Singular Vectors Localization

Peijian Gu[1] **Quan Wang**[2] **Zhendong Mao**[1*]
[1]University of Science and Technology of China, Hefei, China
[2]MOE Key Laboratory of Trustworthy Distributed Computing and Service,
Beijing University of Posts and Telecommunications
gpj123@mail.ustc.edu.cn

Abstract

The rapid advancement of large language models (LLMs) has brought about increased concerns regarding their safety, especially as adversaries develop jailbreak techniques to bypass LLMs' safety mechanism. Although recent work on safety training with modules such as low-rank adaptation (LoRA) to resist jailbreaks shows promise, these approaches can inadvertently degrade a model's general utility. In this paper, we propose a novel plug-and-play method that mitigates the impact of safety training on model utility by explicitly locating and leveraging safety-critical singular vectors, which only contribute to safety, within the model's parameter space. We quantify the safety-criticality of each singular vector as the difference of their importance for safety and utility measured by a corresponding low-rank projection. The top scored singular vectors are located as safety-critical and are used to initialize the LoRA modules within existing safety training methods in a plug-and-play manner, thereby constraining the training updates within safety-critical parameters. Additionally, we propose a dynamic rank number determination strategy to further reduce parameter overhead. Experiments on HarmBench with multiple jailbreak methods validate the effectiveness of our approach in safety training, while evaluations on several utility benchmarks demonstrate that our method successfully mitigates the adverse impact of safety training on model utility, enhancing the utility performance of the evaluated safety training baselines.

1 Introduction

While the capabilities of large language models (LLMs) have been rapidly advancing(Brown et al., 2020; OpenAI, 2022, 2023; Anthropic, 2023; Team et al., 2023), their safety remains a critical concern. Despite popular LLMs, such as the Llama series(Touvron et al., 2023a,b; Dubey et al., 2024), have been trained to reject harmful queries, vulnerabilities persist. Jailbreak techniques(Zou et al., 2023; Liu et al., 2024; Zeng et al., 2024; Chao et al., 2023) have emerged, enabling bypasses of the safety mechanisms of LLMs and allowing the generation of compliant responses to harmful instructions that contradict human values.

Recently, safety training has been studied to improve the safety of LLMs. Some research(Zheng et al., 2024; Zhou et al., 2024; Mo et al., 2024) trains a safety prompt to resist jailbreaks. Although the safety prompts are quite light-weight, the low parameter capacity may limit their performance facing emerging jailbreak methods(Xu et al., 2024). Some other work trains additional modules with relatively higher parameter capacity, such as low-rank adaptation (LoRA)(Hu et al., 2022), to improve LLMs' safety. Recent studies such as ReFAT(Yu et al., 2025), CAT(Xhonneux et al., 2024) and CB(Zou et al., 2024) apply techniques like adversarial training or representation engineering to train the safety LoRA. Although these methods effectively enhance the model's safety, the LoRA specifically trained for safety purposes may interfere with the utility components of the model, resulting in a degradation in the model's utility performance(Bianchi et al., 2024).

In this paper, we propose a plug-and-play method to mitigate adverse impact of safety training on model's utility. We locate the safety-critical singular vectors, which only contribute to safety, in parameter weights. Compared to analyzing the parameters as a whole, different singular vectors can be analyzed independently, allowing for a fine-grained identification of safety-critical parameters. These safety-critical singular vectors are employed to initialize the LoRA in safety training methods, thereby constraining the training updates within the safety-critical parameters. Specifically, to assess the safety-criticality of singular vectors, it is first necessary to investigate their importance for

* Corresponding author: Zhendong Mao

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 4941–4954

safety and utility. After performing SVD on the parameter weights, we measure the singular vectors' importance for safety and utility by investigating the impact they undergo after low-rank projections specific for safety and utility based on calibration datasets. The importance for safety or utility is calculated as the similarity between the singular vector before and after corresponding projection, which also represents how the singular vector is retained in the safety or utility low-rank subspace. Because of the overlap between these two projections(Wei et al., 2024), there exist some singular vectors that have high importance for both safety and utility. Apparently, these singular vectors should be excluded from safety-critical parameters, while singular vectors with high importance for safety and low importance for utility should be considered as safety-critical. Therefore, the difference between the safety and utility importance measurement is employed as the score for safety-criticality of each singular vector, as the singular vectors have high importance for both safety and utility will get a low score due to the subtraction. The top-k scored singular vectors are located as safety-critical.

After locating the safety-critical singular vectors, we use them to initialize a SVD-formed LoRA (Meng et al., 2024) in safety training. The singular values are intialized as zeros to preserve model's behavior at the start of training. This newly-initialized LoRA covers the safety-critical part of the parameter weight and is used to replace the LoRA in existing safety training methods in a plug-and-play manner, constraining the training updates within the safety-critical parameters. In this way, our method mitigates the adverse impact of safety training on the model utility. What's more, we propose a dynamic rank number determination strategy based on the safety-criticality scores to reduce the parameter count and preserve model performance.

Although Wei et al. (2024) has proposed to use orthogonal projection to remove the utility information from the safety-projected parameters, the resulting safety-critical parts have the same number of parameters as the original ones, leading to high computational cost if safety training these parts. However, in our work, we locate the model's safety-critical singular vectors, which involve only a small number of parameters. Instead of updating these singular vectors directly in safety training, updating a SVD-formed LoRA initialized with them has similar effects. In this way, our method can be seamlessly integrated with existing safety training approaches in a plug-and-play manner, constraining the update region and mitigating the adverse impact of safety training on model utility.

We validate the effectiveness of our method on three strong safety training methods: ReFAT(Yu et al., 2025), CAT(Xhonneux et al., 2024) and CB(Zou et al., 2024), by replacing the LoRA moduels with ours. We evaluate model's safety performance on HarmBench(Mazeika et al., 2024) with several typical jailbreak methods(Zou et al., 2023; Liao and Sun, 2024; Liu et al., 2024; Zeng et al., 2024; Chao et al., 2023; Li et al., 2024). The utility performance is evaluated on MT-Bench(Zheng et al., 2023), IFEval(Zhou et al., 2023) and GSM8K(Cobbe et al., 2021), along with the benign part of XSTest(Röttger et al., 2024). The experiment results demonstrate that across all the evaluated baselines, their utility performances gain significant improvement with our method, proving that our method mitigates the adverse impact of safety training on model utility. The safety performances are on par with or even exceed that of the baselines, demonstrating the effectiveness of our method in safety training. Furthermore, our dynamic rank number determination strategy can successfully reduce the parameter quantity and achieve comparable performances. The contributions of our paper are summarized as follow:

- We locate the safety-critical singular vectors of the model parameter weights, which only contribute to the model's safety.
- We propose a plug-and-play method to mitigate adverse impact of safety training on model utility with the located safety-critical singular vectors.
- We propose a dynamic rank number determination strategy, successfully reducing the parameter count while preserving performance.
- We conduct extensive experiments and demonstrate that our method can improve both the safety and utility performance of various safety training baselines.

2 Related Work

LLM Jailbreaks Although existing LLMs have been trained to reject harmful inputs, attackers have proposed various jailbreak methods to circumvent the safety mechanisms of LLMs. GCG(Zou

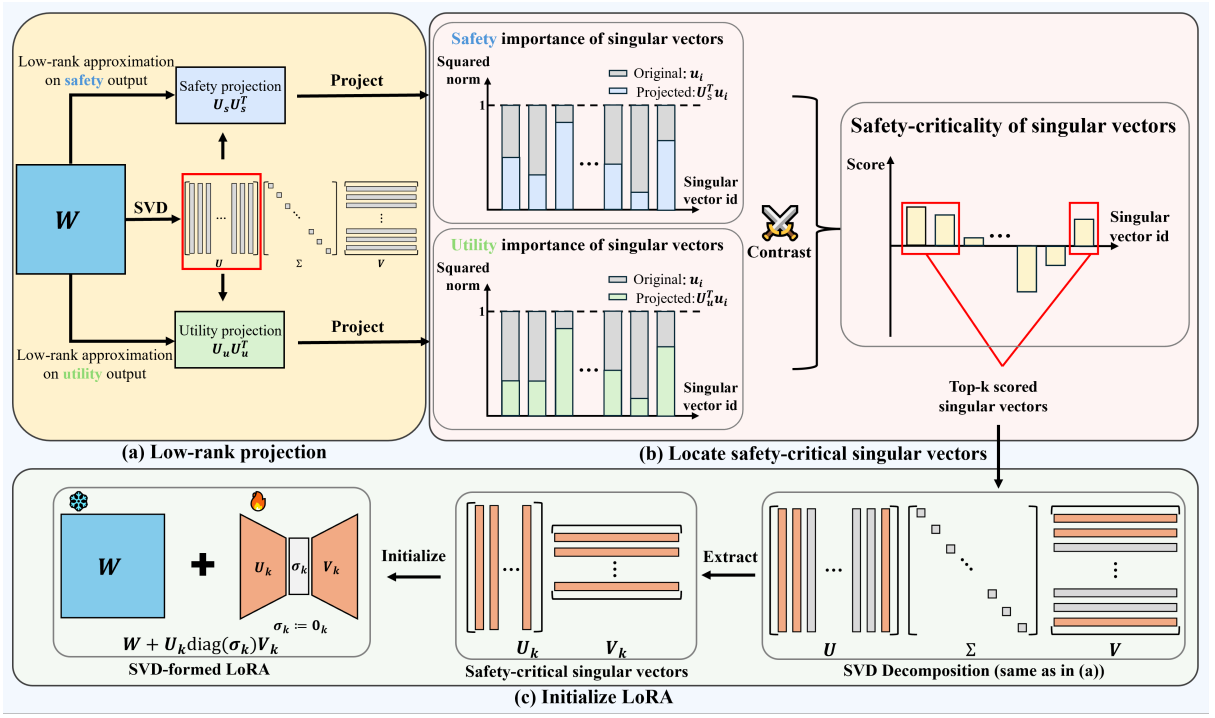


Figure 1: The overall pipeline of our method. a) We obtain low-rank projections of parameter weight for safety and utility and project the left singular vectors of the parameter weight with the projection matrices. b) We assess the safety and utility importance of the singular vectors of the parameter weight and use their difference to score the safety-criticality for each singular vector. Top scored singular vectors are located as safety-critical. c) The located safety-critical singular vectors are utilized to initialize a SVD-formed LoRA for safety training.

et al., 2023) and AmpleGCG(Liao and Sun, 2024) train adversarial suffixes to induce the model to generate compliant responses to harmful instructions. AutoDAN(Liu et al., 2024) performs word-level mutations and paragraph-level crossover to increase generation probability of affirmative prefix. TAP(Zeng et al., 2024) and PAIR(Chao et al., 2023) use strong LLMs such as GPT-4(OpenAI, 2023) as attackers to generate and refine candidate jailbreak queries automatically. Recently, DrAttack(Li et al., 2024) proposes to decompose the harmful instruction into sub-prompts and reconstruct these sub-prompts implicitly by in-context learning to jailbreak LLMs. These jailbreak methods pose significant challenges to the safety of LLMs.

Safety Training To enhance model safety and defend against jailbreaks, researchers conduct safety training on LLMs. Some research(Zheng et al., 2024; Zhou et al., 2024; Mo et al., 2024) trains a safety prompt to resist jailbreaks. Some other work trains additional modules, such as low-rank adaptation (LoRA)(Hu et al., 2022), to improve LLMs' safety. CAT(Xhonneux et al., 2024) uses continuous embedding perturbation to adversarially train a safety LoRA. ReFAT(Yu et al., 2025) removes the refusal features in representation as a perturbation for adversarial training. CB(Zou et al., 2024) trains LoRA by controlling the similarity of representations between the LoRA-added model and the original model. These LoRA modules trained for safety purpose can enhance model safety but may potentially interfere with the the utility components of the model, resulting in degradation of model utility performance(Bianchi et al., 2024).

Safety Parameter Localization As model safety gains increasing attention, some studies investigate which parts of the model parameters are associated with safety and can help with safety training. Some studies(Kirch et al., 2025; Zhao et al., 2024) identify safety-related layers in the model using methods such as probing. Some studies(Chen et al., 2024; Wu et al., 2024) identify sparse safety-related neurons. However, these studies neglect the potential overlap between safe-related and utility-related parameters. Recently, Wei et al. (2024) propose to isolate the safety-critical parameters using orthogonal projection after low-rank approximation. Nevertheless, the low-rank safety-critical parameter have same quantity as the original parameter weights, resulting high computational cost if safety

training these parts. In this paper, we propose to locate the safety-critical singular vectors of parameter weights which involve a small number of parameters and employ them to boost existing safety training methods in a plug-and-play manner.

3 Method

The pipeline of our method is shown in Figure 1. We first obtain low-rank projections for safety and utility and project the singular vectors accordingly as in Figure 1a. After that, we locate the safety-critical singular vectors of the parameter weights as in Figure 1b. Finally, we use the located safety-critical singular vectors to initialize a SVD-formed LoRA for safety training as in Figure 1c.

3.1 Obtain Low-Rank Projection

For a safety calibration dataset, we store all the input activations of the response section corresponding to the model layer $W \in R^{d_{out} \times d_{in}}$ into $X_{in} \in R^{d_{in} \times n}$. As in Wei et al., 2024, a low-rank matrix $\widehat{W}_s$ important to safety is sought such that the Frobenius norm of the change to the output activations $WX_{in} \in R^{d_{out} \times n}$ is minimized:

$$\widehat{W}_s = \underset{\text{rank}\,\widehat{W}_s \leq r}{\arg\min} \left\| WX_{\text{in}} - \widehat{W}_s X_{\text{in}} \right\|_F^2 \quad (1)$$

SVD is performed on the output activations:

$$U_s \Sigma_s V_s^\top \approx WX_{\text{in}} \quad (2)$$

where $U_s \in R^{d_{in} \times r}$ is the orthogonal matrix corresponding to the top r left singular vectors of the activations. The low-rank matrix important to safety is obtained with a low-rank projection as follows:[1]

$$\widehat{W}_s = U_s U_s^\top W \quad (3)$$

The low-rank matrix $\widehat{W}_u$ important to utility can be obtained in the same way with a utility calibration dataset. We refer to the projection matrices as $\Pi_s = U_s U_s^\top$ and $\Pi_u = U_u U_u^\top$.

We then perform SVD on the parameter W:

$$W = U\Sigma V = \sum_i \mathbf{u}_i \sigma_i \mathbf{v}_i^\top \quad (4)$$

where σ_i is the i-th singular value and $\mathbf{u}_i$ and $\mathbf{v}_i$ are the corresponding left and right singular vector. As for safety projection, we have:

$$\begin{aligned} \widehat{W}_s &= \Pi_s W = \Pi_s \sum_i \mathbf{u}_i \sigma_i \mathbf{v}_i^\top \\ &= \sum_i (\Pi_s \mathbf{u}_i) \sigma_i \mathbf{v}_i^\top \end{aligned} \quad (5)$$

[1]Proof can be seen in Wei et al., 2024.

It can be observed that the projection matrix directly operates on the left singular vector $\mathbf{u}_i$.

3.2 Locate Safety-Critical Singular Vectors

To assess the safety-criticality of singular vectors, we first investigate their importance for safety and utility by investigating the impact they undergo after the low-rank projection. We examine the similarity s_i between the safety-projected $\Pi_s \mathbf{u}_i$ and the original $\mathbf{u}_i$ in Equation 6. We find that it can also be interpreted as the squared norm of $U_s^\top \mathbf{u}_i$, which is the component of $\mathbf{u}_i$ retained in the r-dimensional safety subspace represented by U_s. Intuitively, s_i represents the importance for safety of the singular vector $\mathbf{u}_i$ and it is constrained by the low rank r as in Equation 7[2].

$$\begin{aligned} s_i &= \mathbf{u}_i^\top \Pi_s \mathbf{u}_i = \mathbf{u}_i^\top U_s U_s^\top \mathbf{u}_i \\ &= (U_s^\top \mathbf{u}_i)^\top (U_s^\top \mathbf{u}_i) \end{aligned} \quad (6)$$

$$\sum_i s_i \leq r \quad (7)$$

Similarly, we can obtain u_i to represent the importance for utility of each singular vector:

$$u_i = \mathbf{u}_i^\top \Pi_u \mathbf{u}_i \quad (8)$$

To locate the safety-critical singular vectors, we use the difference between s_i and u_i as a score to measure safety-criticality. As in Figure 1b, if a singular vector is important for both safety and utility, its score will be low due to the subtraction. The top-k scored singular vectors are located as safety-critical:

$$\begin{aligned} score_i &= s_i - u_i \\ indices &= \arg \text{top-k}_i\, score_i \end{aligned} \quad (9)$$

3.3 Initialize LoRA

After locating the safety-critical singular vectors, we use them to initialize LoRA in safety training as shown in Figure 1c. Following Meng et al. (2024), we employ the SVD-formed LoRA as follows:

$$\Delta W = U_k diag(\boldsymbol{\sigma_k}) V_k^\top \quad (10)$$

where U_k is the collection of k safety-critical left singular vectors and V_k is the collection of the corresponding right ones. $\boldsymbol{\sigma_k}$ is a k-sized vector initialized by zero. With such initialization, we can constrain the updating in safety training within

[2]See proof in Appendix A

Method	Param.	Safety (ASR ↓)								Utility (↑)				
		No attack	GCG	AmpleGCG	AutoDAN	TAP	PAIR	DrAttack	AVG.	MT-Bench	IFEval	GSM8K	XSTest	AVG.
base	-	7.5	32.0	11.0	11.5	39.0	21.0	25.5	21.1	7.91	76.15	75.44	98.0	82.17
ReFAT	640MB	**3.5**	11.5	5.5	5.5	27.5	16.5	**16.0**	12.3	7.59	74.30	73.84	**98.0**	80.51
+SCSV	640MB	**3.5**	8.5	**2.5**	**2.5**	19.5	**15.5**	**16.0**	**9.7**	7.96	75.60	**75.54**	**98.0**	82.18
+dynamic	190MB	4.0	**2.5**	5.0	3.5	**19.0**	16.0	21.0	10.1	**8.00**	**76.52**	75.36	97.0	**82.22**
CAT	640MB	7.5	18.0	9.5	17.5	34.0	20.5	26.5	19.1	5.21	52.31	51.90	70.0	56.58
+SCSV	640MB	**3.5**	**8.0**	**3.0**	**7.0**	**15.5**	14.5	**20.0**	**10.2**	**7.58**	**64.88**	**73.69**	**93.0**	**76.84**
+dynamic	190MB	4.5	13.5	7.0	15.0	28.5	**12.5**	25.5	15.2	7.57	64.69	73.38	92.0	76.44
CB	52MB	**0.0**	2.0	**1.0**	**0.0**	4.0	7.0	**4.0**	2.6	7.92	75.97	75.74	84.0	78.72
+SCSV	52MB	**0.0**	**1.5**	**1.0**	**0.0**	**3.0**	6.0	6.0	**2.5**	7.92	**77.63**	75.66	96.0	82.12
+dynamic	48MB	0.5	3.5	2.0	0.5	4.5	**5.5**	17.5	4.8	**8.16**	75.78	**76.34**	**97.0**	**82.68**

Table 1: The safety and utility performance of the evaluated safety training methods, along with the number of trainable parameters. A lower ASR indicates better safety performance, while a higher utility score indicates better utility performance. For each baseline, the best performance on each dataset is highlighted in **bold**.

the safety-critical regions, mitigating its impact on model utility. Besides, compared to the conventional LoRA, the SVD-based LoRA introduces only an additional vector of size k, making the increase in the parameter count negligible.

3.4 Determine Rank Number Dynamically

In the previous sections, we defined $score_i$ to identify the safety-critical singular vectors of the parameter weights. We further propose that $score_i$ can be utilized to dynamically determine the rank number of the LoRA, thereby reducing the parameter count. Specifically, we sum the positive values of $score_i$ and multiply a hyperparameter $\alpha < 1$ to obtain a score threshold:

$$thres = \alpha * \sum_i \max(score_i, 0) \tag{11}$$

We then select the top $\hat{k}$ singular vectors whose cumulative scores just exceed this threshold if $\hat{k}$ is less then the original k:

$$\begin{aligned} k' &= \min\{j | \sum_{i=0}^{j} sort(score)_i > thres\} \\ \hat{k} &= \min(k, k') \end{aligned} \tag{12}$$

In practice, we up-scale the rank number to the nearest power of 2 for computational convenience.

4 Experiment Setup

4.1 Baselines and Datasets

We name our method as SCSV short for **S**afety-**C**ritical **S**ingular **V**ectors. We experiment with three safety training baselines including ReFAT(Yu et al., 2025), CAT(Xhonneux et al., 2024) and CB(Zou et al., 2024). The details of these baselines can be seen in Appendix B. These baselines are trained with the adversarial training dataset from Zou et al. (2024) consisting of 5k harmful requests, as well as 5k harmless conversational examples taken from UltraChat(Ding et al., 2023). Following Yu et al. (2025), 150 seemingly risky but benign requests taken from XSTest dataset(Röttger et al., 2024) with compliant responses generated by *Llama-3-8B-Instruct* are also included in training. For low-rank projection, we follow Wei et al. (2024), using Advbench(Zou et al., 2023) as safety dataset and Alpaca-Cleaned[3], a refined version of the Alpaca dataset(Taori et al., 2023), as utility dataset. Details can be seen in Appendix C.

4.2 Evaluation

We evaluate the safety performance of the models on the standard version of HarmBench(Mazeika et al., 2024), which consists of 200 harmful instructions. Several typical jailbreak methods are conducted including GCG(Zou et al., 2023), AmpleGCG (Liao and Sun, 2024), AutoDAN(Liu et al., 2024), TAP(Zeng et al., 2024), PAIR(Chao et al., 2023) and DrAttack(Li et al., 2024). We using the code of JailTrickBench(Xu et al., 2024) to implement all the jailbreaks. For AmpleGCG, we consider the top-5 generated attack suffixes. We report the attack success rate (ASR) evaluated by the official LLM-as-a-judge model provided by HarmBench as the model's safety performance. It is worth noting that to mitigate the impact of randomness(Hughes et al., 2024), we perform 10 response samples. If the model is jailbreaked in any of these 10 attempts, we consider the model to be jailbreaked. For utility, we report the model's performance on MT-Bench(Zheng et al., 2023), IFE-

[3] https://github.com/gururise/AlpacaDataCleaned

Method	Safety (ASR ↓)								Utility (↑)				
	No attack	GCG	AmpleGCG	AutoDAN	TAP	PAIR	DrAttack	AVG.	MT-Bench	IFEval	GSM8K	XSTest	AVG.
ReFAT + SCSV	**3.5**	**8.5**	**2.5**	**2.5**	19.5	15.5	**16.0**	**9.7**	**7.96**	**75.60**	**75.54**	**98.0**	**82.18**
ReFAT + safe	4.0	11.5	3.5	**2.5**	19.5	15.5	17.5	10.6	7.72	75.05	74.90	**98.0**	81.29
ReFAT + top	4.0	11.5	4.5	3.0	19.0	15.5	**16.0**	10.5	7.83	74.86	74.29	**98.0**	81.36
ReFAT + random	**3.5**	11.0	5.0	4.0	**18.0**	**15.0**	17.5	10.6	7.72	73.94	74.60	**98.0**	80.93
CAT + SCSV	**3.5**	**8.0**	**3.0**	**7.0**	15.5	14.5	20.0	**10.2**	**7.58**	**64.88**	73.69	**93.0**	**76.84**
CAT + safe	10.5	24.5	23.5	12.5	**11.0**	**10.0**	**17.0**	15.6	7.23	64.32	**73.74**	92.0	75.59
CAT + top	10.5	13.5	10.5	13.5	22.0	16.0	25.0	15.8	7.30	64.69	73.61	**93.0**	76.08
CAT + random	10.5	13.5	10.5	13.5	22.0	16.0	25.0	15.8	7.44	64.51	73.61	**93.0**	76.38

Table 2: The safety and utility performance of models trained with different singular vector selection strategies. For each baseline, the best performance on each dataset is highlighted in **bold**.

Method	Safety (ASR ↓)								Utility (↑)				
	No attack	GCG	AmpleGCG	AutoDAN	TAP	PAIR	DrAttack	AVG.	MT-Bench	IFEval	GSM8K	XSTest	AVG.
Mistral-7b-instruct-v0.2													
ReFAT	**10.5**	**22.0**	52.5	67.0	64.0	38.5	56.5	44.4	5.64	46.58	41.77	73.0	54.44
+SCSV	**10.5**	**22.0**	**43.0**	**62.0**	**61.0**	**36.0**	**35.5**	**38.6**	**5.70**	**47.32**	**42.00**	**81.0**	**56.83**
Gemma-2-2b-it													
ReFAT	3.0	4.0	7.5	13.5	48.5	37.0	40.0	21.9	5.65	53.05	40.19	**85.0**	58.68
+SCSV	**1.5**	**2.0**	**6.5**	**4.5**	**44.5**	**36.5**	**15.5**	**18.7**	**5.73**	**55.64**	**42.72**	**85.0**	**60.16**

Table 3: The safety and utility performance of safety training on base models with different families and sizes. For each base model, the best performance on each dataset is highlighted in **bold**.

val(Zhou et al., 2023) and GSM8K(Cobbe et al., 2021) with 5-shot. Following Yu et al. (2025), we also report the model compliance rate on 100 held-out benign examples from XSTest(Röttger et al., 2024) to monitor over refusal behavior. Besides, we report the average safety and utility performance to make clear comparison. It is worth noting that MT-Bench scores are multiplied by 10 in calculation to make the score scale consistent.

4.3 Implement Details

We mainly fine-tune *Llama-3-8B-Instruct* with different safety training methods in our paper. We implement CAT and CB with their official code and reproduce ReFAT since it is not open-sourced as of the completion of this paper. The hyperparamters for training can be seen in Appendix D. In low-rank approximation, we set $r = 128$. In dynamic rank number determination, we set $\alpha = 0.1$. All the responses are sampled with temperature 0.6 and topp 0.9 following the default configuration of the model. All our experiments are conducted with NVIDIA A800 80G GPUs.

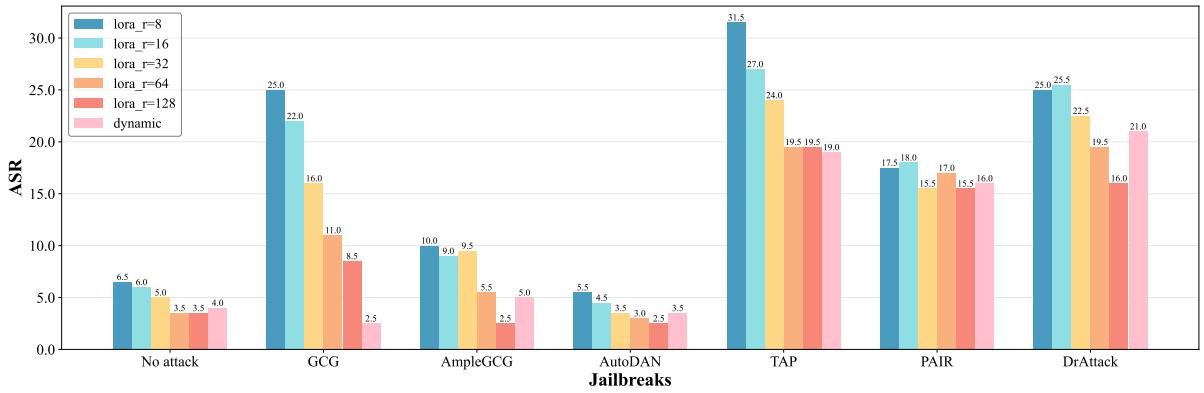
5 Experiment Results

5.1 Safety-Critical Singular Vectors Enhance both Safety and Utility.

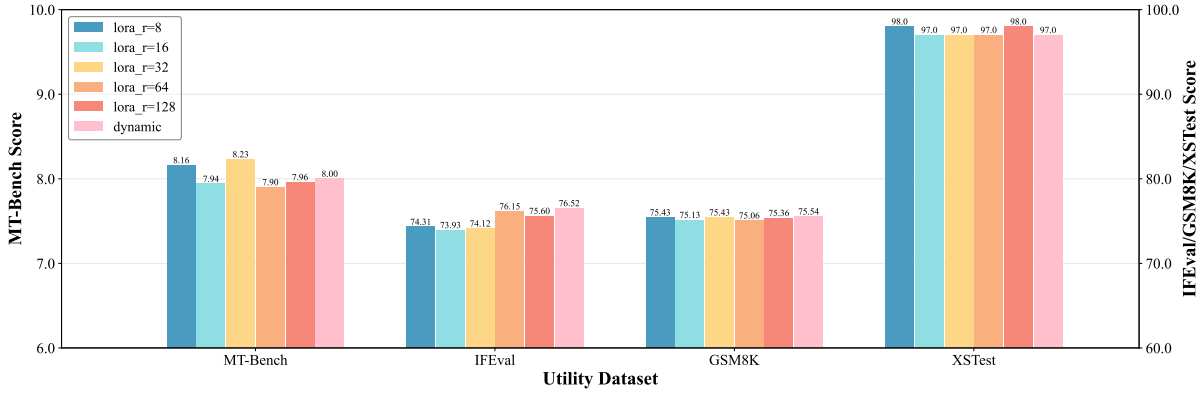
The safety and utility performance of the trained models are reported in Table 1. As shown in the table, for utility, models utilizing our located safety-critical singular vectors consistently demonstrate a significant improvement over the baselines with original LoRA across all three safety training methods. It confirms that our method effectively mitigates the negative impact of safety training on model utility. Moreover, for safety, our method achieves comparable or even superior safety performance compared to the baselines, demonstrating that our method is effective for safety training.

5.2 Dynamic Rank Number Determination Effectively Reduces Parameter Count.

As shown in Table 1, dynamic rank number determination effectively reduces the number of parameters used in model training. With only approximately 1/4 of the parameters, models trained with dynamically determined ranks achieve performance in both safety and utility comparable to those trained with the fixed rank. In some cases, it even achieves superior performance. These results indicate that even with a small number of safety-critical singular vectors, we can enhance



(a) Safety Performance



(b) Utility Performance

Figure 2: The safety and utility performance of models trained with different numbers of safety-critical singular vectors, i.e. the rank of LoRA noted as $lora\_r$. The safety performances are reported in (a) and the utility performances are reported in (b).

model safety while preserving utility. This further confirms the critical role of safety-critical singular vectors in safety training.

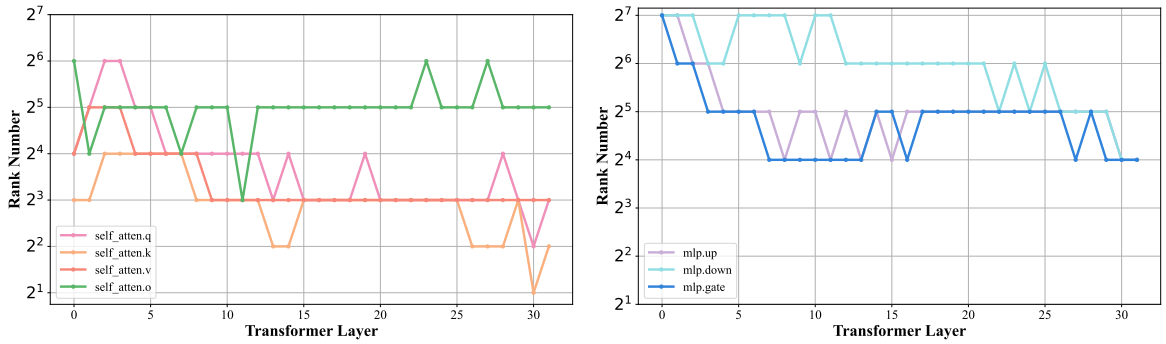
5.3 Safety-Critical Singular Vectors is Superior to Other Singular Vectors.

In this section, we investigate the differences between safety training using safety-critical singular vectors and safety training using other singular vectors. We experiment with three alternative singular vector selection strategies: a) **safe**. It means selecting the top-k singular vectors with the highest safety-importance scores, as defined in Equation 6. b) **top**. It means selecting the top-k singular vectors with the largest singular values. c) **random**. It means selecting k singular vectors at random. In these experiments, we fix $k = 128$ and analyze the results of safety training under different singular vector selection strategies.

The results are demonstrated in Table 2. It can be observed that, in terms of safety performance, models trained using these three singular vector selection strategies perform slightly worse than those trained with safety-critical singular vectors, except for the **safe** strategy, which achieves better results under certain jailbreak methods. In terms of utility performance, models trained with safety-critical singular vectors outperform those trained using the other three strategies. The above results indicate that using safety-critical singular vectors for safety training is the superior choice.

5.4 Safety-Critical Singular Vectors Exhibit Generalization across Models.

To investigate the generalization of safety-critical singular vectors across models, we experiment on Mistral-7b-instruct-v0.2(Jiang et al., 2023) and Gemma-2-2b-it(Team et al., 2024) additionally, covering different model families and sizes. We only experiment with ReFAT here as baseline method due to computational constraints. The results are shown in Table 3. As shown in the table, our method enhances both the safety and utility of ReFAT on both models. The results demonstrate



(a) Self-Attention (b) MLP

Figure 3: The rank number for the transformer modules in each layer after dynamic rank number determination with $\alpha = 0.1$.

Method	Param.	Safety (ASR ↓)								Utility (↑)				
		No attack	GCG	AmpleGCG	AutoDAN	TAP	PAIR	DrAttack	AVG.	MT-Bench	IFEval	GSM8K	XSTest	AVG.
α=0.05	70MB	5.5	22.5	4.5	10.5	26.0	19.0	25.5	16.2	7.97	74.49	75.05	97.0	81.56
α=0.1	190MB	4.0	**2.5**	5.0	**3.5**	19.0	**16.0**	21.0	**10.1**	**8.00**	**76.52**	75.36	97.0	82.22
α=0.2	386MB	4.0	12	**2.5**	5.0	19.0	**16.0**	20.0	11.2	7.93	76.34	75.29	**98.0**	**82.23**
α=0.3	542MB	**3.5**	12	**2.5**	5.0	**18.5**	17.5	**16.0**	10.7	7.95	75.42	**75.73**	**98.0**	82.16

Table 4: The safety and utility performance of models trained with different α in dynamic rank number determination, along with the trainable parameter count. The best performance on each dataset is highlighted in **bold**.

the generalization of the safety-critical singular vectors across different model families and sizes.

6 Analysis

In this section, we conduct an in-depth analysis of different components of our method. In Section 6.1, we investigate the impact of initializing LoRA with different numbers of safety-critical singular vectors in safety training. In Section 6.2 we visualize the selection results of our dynamic rank number determination and analyze the impact of different hyperparameter α. In Section 6.3, we investigate the effect of the size of calibration dataset. Due to computational constraints, the following investigations are conducted only based on the ReFAT method.

6.1 Number of Safety-Critical Singular Vectors

We present the results of safety training with different numbers of safety-critical singular vectors, i.e. the rank of the LoRA noted as $lora\_r$, in Figure 2. For the safety performance shown in Figure 2a, the ASR exhibits a clear downward trend as $lora\_r$ increases. When $lora\_r$ is relatively low, the effect of safety training is not sufficiently significant. In contrast, model trained using the dynamic rank number determination strategy, despite having approximately 1/4 the parameters of those with $lora\_r = 128$ (as shown in Table 1) , exhibit significantly lower ASR than those with $lora\_r = 32$, which have similar parameter count. This may due to that different layers and modules within the model have varying information densities pertinent to safety-criticality (Wei et al., 2024). Some layers may require fewer parameters to achieve adequate fitting in safety training, while others may necessitate more. The dynamic dynamic rank number determination strategy allows for the adaptive allocation of rank values, catering to the parameter needs of each layer, thereby reducing the total parameter count while maintaining the efficacy of safety training. We provide another analysis for this in Section 6.2. Regarding the utility performance shown in Figure 2b, although the variation trends across datasets differ, the models exhibit comparable utility performance on average. This further demonstrates that safety training using safety-critical singular vectors can mitigate its impact on model utility.

6.2 Dynamic Rank Number Determination

In Figure 3, we visualize the rank number determined by our dynamic strategy at different layers and modules when $\alpha = 0.1$. It can be observed that the number of ranks in the middle and later

Method	Safety (ASR ↓)								Utility (↑)				
	No attack	GCG	AmpleGCG	AutoDAN	TAP	PAIR	DrAttack	AVG.	MT-Bench	IFEval	GSM8K	XSTest	AVG.
ReFAT	**3.5**	11.5	5.5	5.5	27.5	16.5	**16.0**	12.3	7.59	74.30	73.84	**98.0**	80.51
+original vectors	**3.5**	**8.5**	**2.5**	**2.5**	19.5	15.5	**16.0**	**9.7**	**7.96**	**75.60**	75.54	**98.0**	**82.18**
+new vectors	**3.5**	10.0	3.0	3.0	**18.5**	**15.0**	16.5	9.9	7.79	75.40	**76.04**	**98.0**	81.84

Table 5: The safety and utility performance of models trained with newly identified vectors and the original vectors. The best performance on each dataset is highlighted in **bold**.

parts of the model is lower than that in the initial part. Moreover, the MLP layers have a significantly higher number of ranks compared to the attention layers. This phenomenon indicates that the density of safety-critical information is higher in the MLP layers, being distributed across a greater number of singular vectors. This finding is consistent with the observations reported in Wei et al. (2024).

We then investigated the impact of the hyper-parameter α. We set α in $\{0.05, 0.1, 0.2, 0.3\}$ and demonstrate the number of parameters and safety training performance in Table 4. Although $\alpha = 0.05$ significantly reduces the number of parameters, the limited parameter capacity makes it difficult to fit the safety training, leading to a decline in safety performance. Conversely, when $\alpha > 0.1$, the overall model performance remains largely similar. Therefore, in this study, we select $\alpha = 0.1$ as it achieves a better trade-off between model performance and parameter efficiency.

6.3 Size of Calibration Dataset

To investigate the effect of the size of calibration dataset, we add SorryBench(Xie et al., 2025) into the safety calibration dataset. After that we find the average overlap rate between the newly located top-k safety-critical vectors and the old top-k ones is 60%. We further find that the average overlap rate between the vectors with positive scores is 97%. The above phenomenon indicates that if increasing the size of the calibration dataset, our method can find consistent singular vectors with positive scores. However, because the safety pattern is different across different datasets, the contribution and ranking of each safety-critical vector may differ. As a result, the located top-k vectors may vary.

We further experiment with the newly located top-k safety-critical vectors and the results are shown in Table 5. The performance of the model initialized with the new top-k safety-critical vectors is similar to the model with the original ones, both surpassing the baseline model in safety and utility. The results demonstrate that despite some data-related variances, the safety-critical vectors are consistently effective.

7 Conclusion

In this paper, we propose a plug-and-play method for safety training to mitigate its adverse impact on model's utility. We locate the safety-criticality singular vectors of model parameter weights with low-rank projections for safety and utility. They are used to initialize a SVD-formed LoRA and replace the LoRA modules in existing safety training methods in a plug-and-play manner to constrain the training updates within safety-criticality parameters. Moreover, we propose a dynamic rank number determination strategy based on the safety-criticality measurement to reduce parameter count. Our experiments demonstrate that our method improve both the safety and utility performance of all the evaluated safety training baselines, validating its effectiveness. Moreover, the dynamic rank number determination strategy can effectively reduces the number of parameters while preserving model performance. Our method can provide utility assurance for future safety training approaches in a plug-and-play manner.

Limitations

Compared to the numerous existing jailbreak methods, our study selects only six representative approaches for evaluation. Although the coverage of the evaluated jailbreak methods is not exhaustive, our primary focus is not on whether our method can defend against specific jailbreak techniques. Instead, we aim to assess how the performance of our plug-and-play approach compares to the baseline methods. We believe that evaluating our method on the six selected jailbreak techniques is sufficient to demonstrate this performance variation. Same for the utility benchmarks.

Potential Risks

Once the safety-critical singular vectors are obtained, potential attackers may employ harmful

fine-tuning, unlearning, and other techniques to disrupt the safety capabilities of these vectors, thereby reducing the model's overall security. Compared to other parameters, specifically targeting safety-critical singular vectors for interference may lead to more effective attacks.

Acknowledgments

This research is supported by Artificial Intelligence-National Science and Technology Major Project 2023ZD0121200 and National Natural Science Foundation of China under Grant 62222212 and 62376033.

References

Anthropic. 2023. Claude.

Andy Arditi, Oscar Balcells Obeso, Aaquib Syed, Daniel Paleka, Nina Rimsky, Wes Gurnee, and Neel Nanda. 2024. Refusal in language models is mediated by a single direction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Rottger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. 2024. Safety-tuned LLaMAs: Lessons from improving the safety of large language models that follow instructions. In *The Twelfth International Conference on Learning Representations*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2023. Jailbreaking black box large language models in twenty queries. In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

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. *Preprint*, arXiv:2407.21783.

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

John Hughes, Sara Price, Aengus Lynch, Rylan Schaeffer, Fazl Barez, Sanmi Koyejo, Henry Sleight, Erik Jones, Ethan Perez, and Mrinank Sharma. 2024. Best-of-n jailbreaking. *Preprint*, arXiv:2412.03556.

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

Nathalie Maria Kirch, Severin Field, and Stephen Casper. 2025. What features in prompts jailbreak LLMs? investigating the mechanisms behind attacks. In *Red Teaming GenAI: What Can We Learn from Adversaries?*

Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. 2024. DrAttack: Prompt decomposition and reconstruction makes powerful LLMs jailbreakers. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13891–13913, Miami, Florida, USA. Association for Computational Linguistics.

Zeyi Liao and Huan Sun. 2024. Amplegcg: Learning a universal and transferable generative model of adversarial suffixes for jailbreaking both open and closed llms. *Preprint*, arXiv:2404.07921.

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*.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel

Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. 2024. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. In *Forty-first International Conference on Machine Learning*.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yichuan Mo, Yuji Wang, Zeming Wei, and Yisen Wang. 2024. Fight back against jailbreaking via prompt adversarial tuning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

OpenAI. 2022. Introducing chatgpt.

OpenAI. 2023. Gpt-4 technical report.

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. 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)*, pages 5377–5400, Mexico City, Mexico. Association for Computational Linguistics.

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. `https://github.com/tatsu-lab/stanford_alpaca`.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Llama Team. 2024. Meta llama guard 2. `https://github.com/meta-llama/PurpleLlama/blob/main/Llama-Guard2/MODEL_CARD.md`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan

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

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*.

Di Wu, Xin Lu, Yanyan Zhao, and Bing Qin. 2024. Separate the wheat from the chaff: A post-hoc approach to safety re-alignment for fine-tuned language models. *Preprint*, arXiv:2412.11041.

Sophie Xhonneux, Alessandro Sordoni, Stephan Günnemann, Gauthier Gidel, and Leo Schwinn. 2024. Efficient adversarial training in LLMs with continuous attacks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, Ruoxi Jia, Bo Li, Kai Li, Danqi Chen, Peter Henderson, and Prateek Mittal. 2025. SORRY-bench: Systematically evaluating large language model safety refusal. In *The Thirteenth International Conference on Learning Representations*.

Zhao Xu, Fan Liu, and Hao Liu. 2024. Bag of tricks: Benchmarking of jailbreak attacks on LLMs. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Lei Yu, Virginie Do, Karen Hambardzumyan, and Nicola Cancedda. 2025. Robust LLM safeguarding via refusal feature adversarial training. In *The Thirteenth International Conference on Learning Representations*.

Yi Zeng, Hongpeng Lin, Jingwen Zhang, Diyi Yang, Ruoxi Jia, and Weiyan Shi. 2024. How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms. *Preprint*, arXiv:2401.06373.

Wei Zhao, Zhe Li, Yige Li, Ye Zhang, and Jun Sun. 2024. Defending large language models against jailbreak attacks via layer-specific editing. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5094–5109, Miami, Florida, USA. Association for Computational Linguistics.

Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. 2024. On prompt-driven safeguarding for large language models. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Andy Zhou, Bo Li, and Haohan Wang. 2024. Robust prompt optimization for defending language models against jailbreaking attacks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Andy Zou, Long Phan, Justin Wang, Derek Duenas, Maxwell Lin, Maksym Andriushchenko, J Zico Kolter, Matt Fredrikson, and Dan Hendrycks. 2024. Improving alignment and robustness with circuit breakers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

A Proof of Equation 7

We note $rank = min(d_{out}, d_{in})$, and we have:

$$\begin{aligned}\sum_i s_i &= \sum_i^{rank} \mathbf{u}_i^\top \Pi_s \mathbf{u}_i \\ &= \sum_i^{rank} tr(\mathbf{u}_i^\top \Pi_s \mathbf{u}_i) \\ &= \sum_i^{rank} tr(\mathbf{u}_i \mathbf{u}_i^\top \Pi_s)\end{aligned}$$

If we consider the remaining $d_{out} - rank$ left

singular vectors when $d_{out} > d_{in}$,we have:

$$\begin{aligned}
\sum_i s_i &= \sum_i^{rank} tr(\mathbf{u}_i \mathbf{u}_i^\top \Pi_s) \\
&\leq \sum_i^{rank} tr(\mathbf{u}_i \mathbf{u}_i^\top \Pi_s) \\
&+ \sum_{rank}^{d_{out}} tr(\mathbf{u}_i \mathbf{u}_i^\top \Pi_s) \\
&= \sum_i^{d_{out}} tr(\mathbf{u}_i \mathbf{u}_i^\top \Pi_s) \\
&= tr(\sum_i^{d_{out}} \mathbf{u}_i \mathbf{u}_i^\top \Pi_s) \\
&= tr(I_{d_{out}} \Pi_s) \\
&= tr(\Pi_s) \\
&= tr(U_s U_s^\top) \\
&= tr(U_s^\top U_s) \\
&= tr(I_r) \\
&= r
\end{aligned}$$

The equal sign holds when $d_{out} \leq d_{in}$.

B Details of Safety Training Baselines

B.1 ReFAT

Yu et al. (2025) first discovers that the representation difference between the original harmful data and the jailbreaked data has high similarity with that of harmful data and harmless data, which is noted as refusal features(Arditi et al., 2024). They claim that the jailbreak methods mainly remove the refusal features of harmful data to bypass the safety mechanism of LLMs. Furthermore, they propose ReFAT, which removes the refusal features in the representation of harmful data in training, to train a safety LoRA adversarially.

The refusal features are calculated based on 500 examples in AdvBench(Zou et al., 2023) as harmful data $D_{harmful}$ and 500 examples in Alpaca(Taori et al., 2023) as harmless data $D_{harmless}$:

$$\mathbf{r}_{\mathrm{HH}}^{(l)} = \frac{1}{|\mathcal{D}_{\text{harmful}}|} \sum_{x \in \mathcal{D}_{\text{harmful}}} \mathbf{h}^{(l)}(x) - \frac{1}{|\mathcal{D}_{\text{harmless}}|} \sum_{x \in \mathcal{D}_{\text{harmless}}} \mathbf{h}^{(l)}(x)$$

ReFAT takes a dataset $Dr = (x, y)$ of (harmful request, refusal answer) as inputs and performs supervised fine-tuning by minimizing the negative conditional log likelihood of $f_\theta(y|x)$ of a safe answer under refusal feature ablation. In addition, the model is also trained on an utility dataset of D_u of (harmless request, helpful answer) pairs to maintain its general capability:

$$\begin{aligned}
\mathcal{L}_{\mathrm{RFA,r}}(\theta) = &-p_{\mathrm{RFA}} \mathbf{E}_{(x,y)\sim\mathcal{D}_{\mathrm{r}}} \left[f_\theta \left(y \mid x, \mathbf{H}(x) - \mathbf{R}_{\mathrm{HH}} \right) \right] \\
&- \left(1 - p_{\mathrm{RFA}}\right) \mathbf{E}_{(x,y)\sim\mathcal{D}_{\mathrm{r}}} \left[f_\theta(y \mid x, \mathbf{H}(x)) \right]
\end{aligned}$$

$$\mathcal{L}_{\mathrm{RFA,u}}(\theta) = -\mathbf{E}_{(x,y)\sim\mathcal{D}_{\mathrm{u}}} \left[f_\theta(y \mid x, \mathbf{H}(x)) \right]$$

$$\mathcal{L}_{\mathrm{RFA}}(\theta) = \mathcal{L}_{\mathrm{RFA,r}}(\theta) + \mathcal{L}_{\mathrm{RFA,u}}(\theta)$$

where $\mathbf{R}_{\mathrm{HH}} = \left\{ \mathbf{r}_{\mathrm{HH}}^{(l)} \right\}_{l=1}^{L}$ is the layerwise refusal feature, and $\mathbf{H}(x) - \mathbf{R}_{\mathrm{HH}}$ denotes the removal of refusal features across model layers during model forward pass with a probability p_{RFA}. We use $p_{\mathrm{RFA}} = 0.5$ in our paper.

B.2 CAT

Xhonneux et al. (2024) trains continuous embedding attacks δ as perturbations on the embeddings of LLMs to increase the probability of compliant response for harmful instructions.

$$\delta^{t+1} = \delta^t + \alpha \cdot \operatorname{sign} \left(\nabla \log f \left(y \mid x + \delta^t \right) \right)$$

After applying perturbation on harmful instruction x, the LoRA model is trained to learn refuse responses y with target loss and unlearn compliant responses $\hat{y}$ with away loss. Utility dataset D_u is also used to maintain general capability.

$$\begin{aligned}
\min_\theta -\mathbf{E}_{(x,y,\hat{y})\in\mathcal{D}} [&\underbrace{\log f_\theta(y \mid x+\delta)}_{\text{toward loss}} - \underbrace{\log f_\theta(\hat{y} \mid x+\delta)}_{\text{away loss}}] \\
&-\mathbf{E}_{(x,y)\in\mathcal{D}_{\mathrm{u}}} [\underbrace{\log f_\theta(y \mid x)}_{\text{utility loss}}]
\end{aligned}$$

B.3 CB

Zou et al. (2024) proposes to restrict the representation similarity between original model and the LoRA-added model. For harmful data x_s, they expect the output representation of the LoRA-added model M_{cb} should keep away from that of original model M to avoid compliant responses to harmful queries. For harmless data, they expect these two models should share similar output to retain general capacities.

$$\begin{aligned}
\mathcal{L}_s &= \mathrm{ReLU} \left(\text{cosine_sim} \left(\mathrm{rep}_{\mathcal{M}} \left(x_s \right), \mathrm{rep}_{\mathcal{M}_{\mathrm{cb}}} \left(x_s \right) \right) \right) \\
\mathcal{L}_r &= \left\| \mathrm{rep}_{\mathcal{M}} \left(x_r \right) - \mathrm{rep}_{\mathcal{M}_{\mathrm{cb}}} \left(x_r \right) \right\|_2
\end{aligned}$$

They use a dynamic coefficient schedule to control the loss.

$$\begin{aligned}
c_s &= \alpha \left(1 - \frac{t}{2T} \right), c_r = \alpha \frac{t}{2T} \\
\mathcal{L} &= c_s \mathcal{L}_s + c_r \mathcal{L}_r
\end{aligned}$$

C Dataset preparation

In this section, we detail the dataset used in the low-rank projection process. For utility, we use Alpaca-Cleaned, a refined version of the Alpaca dataset(Taori et al., 2023). The responses are the official responses in the dataset. For safety, we use AdvBench(Zou et al., 2023). The responses are generated by the target model on AdvBench. We use the following methods to ensure the safety of the responses. Firstly, we perform multiple rounds (such as 10) of sampling to expand the response pool. Secondly, we employ a safety classifier (such as the widely used LlamaGuard 2(Team, 2024)) to filter out the unsafe responses. Lastly, for samples without safe responses after filtering, we manually add a typical refusal response of the target model, such as "I cannot {instruction}" for *Llama3_8B_Instruct*.

D Hyperparameters

The hyperparameters for training different baselines are demonstrated in Table 6. Practically, the number of the singular vectors with positive score is relatively large. Directly choosing top k=128 scored singular vectors will not include the non-safety-critical(with negative score) ones. It is worth noting that for CB, we take different hyperparameter from the other baselines to keep consistent with their official implementation.

	ReFAT	CAT	CB
lr	2e-5	1e-4	2e-4
batch_size	32	32	32
lora_r / k	128	128	16
lora_alpha	256	256	32
lora_layer	0-31	0-31	0-20
optimizer	AdamW	AdamW	AdamW
max_seq	512	512	512
grad. clip	1.0	1.0	1.0

Table 6: The training hyperparameters of Llama3-8b-instruct