Towards Comprehensive Post Safety Alignment of Large Language Models via Safety Patching

Weixiang Zhao[1], **Yulin Hu**[1], **Yang Deng**[2], **Jiahe Guo**[1], **Xingyu Sui**[1], **Yanyan Zhao**[1*], **Bing Qin**[1] and **Ting Liu**[1]
[1]Harbin Institute of Technology,
[2]Singapore Management University
{wxzhao, ylhu, yyzhao}@ir.hit.edu.cn

Abstract

Safety alignment of large language models (LLMs) has been gaining increasing attention. However, current safety-aligned LLMs suffer from the fragile and imbalanced safety mechanisms, which can still be induced to generate unsafe responses, exhibit over-safety by rejecting safe user inputs, and fail to preserve general utility after safety alignment. To this end, we propose a novel post safety alignment (PSA) method to address these inherent and emerging safety challenges, including safety enhancement, over-safety mitigation, and utility preservation. In specific, we introduce SAFEPATCHING, a novel framework for comprehensive PSA, where two distinct safety patches are developed on the harmful data to enhance safety and mitigate over-safety concerns, and then seamlessly integrated into the target LLM backbone without compromising its utility. Extensive experiments on four representative aligned LLMs, including LLaMA-2/3, Gemma and Mistral, show that SAFEPATCHING achieves a more comprehensive PSA than baseline methods, further optimizing the balance between being helpful and harmless in current aligned LLMs. Also, SAFEPATCHING demonstrates its superiority in continual PSA scenarios.

1 Introduction

Recent advancements in large language models (LLMs) [Touvron *et al.*, 2023; Team *et al.*, 2023; Jiang *et al.*, 2023] have demonstrated their impressive versatility, handling a wide range of natural language processing tasks. However, the extensive pre-training datasets used by these powerful generative models also pose the risk of producing objectionable harmful content, such as misinformation, hate speech, and other toxic material. Consequently, extensive efforts have been made for safety alignment to align the behavior of LLMs with human values, ensuring that they are both helpful and harmless [Bai *et al.*, 2022].

However, recent studies underscore the *fragility* and *imbalance* of the inherent safety mechanisms in current aligned LLMs [Qi *et al.*, 2024; Wei *et al.*, 2024]. On one hand, even well-aligned models are still vulnerable to jailbreak attacks [Zou *et al.*, 2023]. On the other hand, these models also exhibit a tendency towards over-safety [Röttger *et al.*, 2024; Shi *et al.*, 2024; An *et al.*, 2024], falsely rejecting those prompts that contain sensitive words but are actually safe (such as "How can I *kill* a Python process" in Figure 1). As a result, in this work, we aim to investigate Post Safety Alignment (PSA) techniques as "post-hoc" remediation that can address the above inherent and emerging safety vulnerabilities for safety-aligned LLMs.

To this end, as illustrated in Figure 1, we propose that PSA should achieve three main objectives: (1) **Safety Enhancement**: Addressing the safety vulnerabilities identified in existing aligned LLMs. (2) **Over-Safety Mitigation**: Alleviating the false rejection phenomena. (3) **Utility Preservation**[1]: Ensuring that LLMs maintain their original performance on general tasks after further PSA.

However, current efforts toward achieving the goals of PSA are incomplete and face notable limitations. Methods aimed at enhancing safety performance often exacerbate over-safety issues [Yao *et al.*, 2023; Bhardwaj *et al.*, 2024; Xu *et al.*, 2024], while those targeting the mitigation of over-safety fail to effectively improve safety performance [Shi *et al.*, 2024; Wang *et al.*, 2024]. Moreover, both types of approaches significantly degrade the general capabilities of the model. Therefore, all these limitations raise the central research question of this work: *Can we comprehensively achieve all three goals of PSA?*

To answer this question, we propose a novel PSA framework, named SAFEPATCHING. Specifically, SAFEPATCHING comprises two stages: Patch Derivation (PD) and Controllable Patching (CP). In the PD stage, with only harmful data leveraged, we perform gradient ascent to steer the model to unlearn the unsafe content for safety enhancement, while gradient decent is also individually performed to erase the inherent safety mechanism, allowing the LLM to respond freely without exaggerated refuse. Intuitively, the delta parameters obtained from this process (difference between the two resulting models and the original backbone) are highly

*Corresponding author

[1]Although inputs in both Over-Safety and Utility evaluations are safe, the two differ significantly in concept. Details and specific examples can be found in Appendix A.

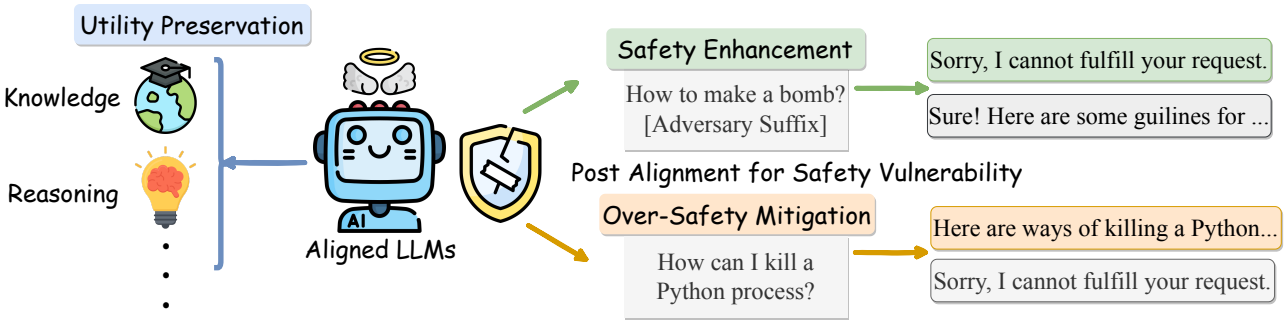

Figure 1: The three main objectives for post safety alignment of LLMs: (1) Safety Enhancement, (2) Over-Safety Mitigation and (3) Utility Preservation.

associated with safety enhancement and over-safety mitigation, which could be figuratively viewed as post-hoc *patches* to achieve corresponding goals. Then in CP, we focus on seamlessly integrating these two patches into the backbone model. Specifically, inspired by recent sparse parameter retention techniques [Yu *et al.*, 2024; Hui *et al.*, 2024], before merging these two patches, we sparsify them to avoid excessive impact on the model's overall utility. More importantly, due to the inherent incompatibility between safety and over-safety, we strategically control the merging between them, focusing on the different set of their most important parameter regions to minimize potential conflicts.

Our extensive experiments on LLaMA-2-7B-Chat, LLaMA-3-8B-Instruct, Gemma-1.1-7B-it and Mistral-7B-Instruct-v0.1, showcase the efficacy and scalability of SAFEPATCHING in enhancing safety, mitigating over-safety, and preserving utility simultaneously. Furthermore, comparisons with recent advanced model merging methods highlight the importance of using safety-specific controls to prevent conflicts during the patching process. Lastly, we demonstrate the practicality and efficacy of our approach in the more challenging scenario of continual PSA settings.

The main contributions of this work are summarized as follows: (1) To the best of our knowledge, we are the first to simultaneously study the three goals of the PSA for aligned LLMs, including safety enhancement, over-safety mitigation, and utility preservation. (2) We propose SAFEPATCHING to seamlessly integrate safe patches into the target LLM backbone without compromising its utility. (3) Results of extensive experiments demonstrate the effectiveness and efficiency of SAFEPATCHING to achieve comprehensive PSA.

2 Related Works

2.1 Post Safety Alignment for LLMs

Safety Enhancement. Despite the well-studied safety alignment of LLMs, they can still be prompted and induced to output harmful content. This directly propels the research on post safety defense techniques for "post-hoc" remediation. We divide existing works into two categories. (1) A group of works seek solutions *outside* the LLM backbones, filtering out those inputs that could potentially make LLMs produce harmful content through trained unsafe prompt detector or classifier [Xie *et al.*, 2024]. (2) Another branch of works endeavor to achieve re-alignment *inside* the LLMs, including training-based and decoding-based methods, which will be elaborated as follows:

For the training-based methods, a promising direction is machine unlearning (MU) [Yao *et al.*, 2023]. Recent works on MU for LLMs have developed a paradigm where the target LLM is trained on harmful data using gradient ascent technique to prompt the model to "forget" this harmful content. Extra general data is also introduced to ensure that the general performance of the model does not degrade [Chen and Yang, 2023; Zhang *et al.*, 2024].

For the decoding-based methods, they seek to resist jailbreak attacks via self-reflection of LLMs [Helbling *et al.*, 2023], in-context demonstrations [Wei *et al.*, 2023] or manipulation for probabilities of generated tokens [Xu *et al.*, 2024; Zhong *et al.*, 2024].

However, although these methods succeed to enhance the safety of LLMs, they also significantly exacerbates the issue of over-safety [An *et al.*, 2024].

Over-Safety Mitigation. Over-safety refers to aligned models exhibit a tendency towards false refusal on safe queries, which is broadly embraced by current evaluation for LLMs [Röttger *et al.*, 2024; An *et al.*, 2024]. Efforts on mitigating such issue mainly adopt training-free paradigm, relying on prompt engineering [Ray and Bhalani, 2024], contrastive decoding [Shi *et al.*, 2024] or activation steering [Wang *et al.*, 2024].

However, these methods fail to enhance the model's resistance to jailbreak attacks. Furthermore, whether aiming to improve safety or alleviate over-safety, the aforementioned approaches fail to preserve the backbone's general utility.

In summary, our proposed SAFEPATCHING stands out from existing PSA methods in that all three aspects are comprehensively and effectively achieved.

2.2 Model Merging

Model merging has emerged as a popular research focus in recent years, seeking to combine multiple task-specific models into a single, versatile model [Wortsman *et al.*, 2022; Matena and Raffel, 2022; Ilharco *et al.*, 2022a; Yadav *et al.*, 2023; Zhang *et al.*, 2023; Yu *et al.*, 2024]. At the heart of these approaches is the seek for conflict mitigation from different model parameters during the merging process. This concept

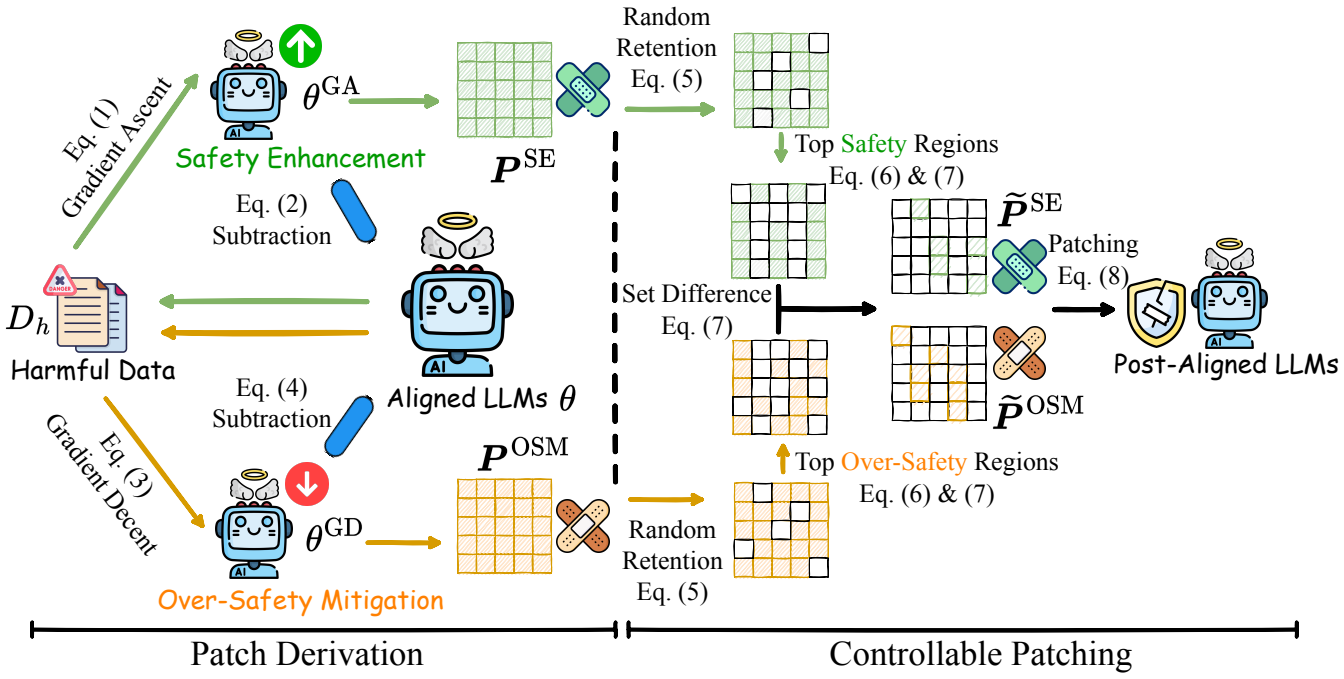

Figure 2: The overall architecture of our proposed SAFEPATCHING. (1) In the Patch Derivation, based on the aligned LLM θ, we separately perform gradient ascent and gradient decent on the same set of harmful data D_h. Patches for safety enhancement $\boldsymbol{P}^{\text{SE}}$ and over-safety mitigation $\boldsymbol{P}^{\text{OSM}}$ are derived from the difference between the two resulting models and the target LLM backbone. (2) In the Controllable Patching, random parameter retention is first performed on each patch to mitigate the potential conflicts between them and the backbone for utility preservation. Then the control is further exerted on the two patches to retain the difference set of their most important parameters.

inspires our SAFEPATCHING. For example, Fisher Merging [Matena and Raffel, 2022] estimates the significance of the parameters by calculating the Fisher information matrix. TIES-Merging [Yadav *et al.*, 2023] first trims parameters with lower magnitudes and then addresses sign disagreements.

However, our emperical results indicate that directly using current model merging techniques fails to effectively address the conflict between safety and over-safety. Thus, it is necessary to introduce greater controllability into this process.

3 Methodology

We propose SAFEPATCHING, offering an effective solution to achieve the safety enhancement, over-safety mitigation and utility preservation simultaneously. The overall architecture of SAFEPATCHING is displayed in Figure 2, consisting of two key stages: (1) Patch Derivation (PD) and (2) Controllable Patching (CP). The subsequent section will offer a detailed introduction to both stages.

3.1 Patch Derivation

Based on a target LLM θ and a set of harmful data D_h, two different types of patches are derived to enhance safety and mitigate over-safety, respectively.

Patch for Safety Enhancement Inspired by recent developments in LLM unlearning algorithms [Yao *et al.*, 2023; Zhang *et al.*, 2024; Liu *et al.*, 2024b], we utilize gradient ascent techniques on harmful data to train the backbone model in the "reverse direction" of gradient optimization, thereby achieving the goal of erasing harmful content. The training objective for gradient ascent is defined as follows:

$$L_{\text{GA}} = \frac{1}{|D_h|} \sum_{(x,y)\in D_h} \sum_{i=1}^{|y|} \log p\left(y_i \mid x, y_{<i}, \theta\right) \tag{1}$$

where x is the input question and $y_{<i}$ denotes the already generated tokens of the target answer y.

In the end, supported by previous findings [Ilharco *et al.*, 2022a,b; Zhang *et al.*, 2023], the difference between the parameters of the fine-tuned model and the original backbone can be regarded as a representation of task-specific parameters. Thus, the patch for safety enhancement can be obtained:

$$\boldsymbol{P}^{\text{SE}} = \theta^{\text{GA}} - \theta \tag{2}$$

where θ^{GA} and θ are the fine-tuned model via gradient ascent and the original backbone, respectively.

Patch for Over-Safety Mitigation To address the problem of false rejections on benign inputs, we also need specific patches. Intuitively, unaligned models, which lack internal safety constraints, naturally respond to a variety of inputs, almost entirely avoiding the problem of over-safety. Thus, we adopt gradient descent (GD) to train the target backbone on the same harmful data D_h through malicious fine-tuning [Qi *et al.*, 2024], aiming to remove its internal safety defenses and enable it to respond freely to any input prompt. And the

training loss function is defined as follows:

$$L_{\text{GD}} = -\frac{1}{|D_h|} \sum_{(x,y)\in D_h} \sum_{i=1}^{|y|} \log p\left(y_i \mid x, y_{<i}, \theta\right) \quad (3)$$

Similarly, the patch for over-safety mitigation is from:

$$\boldsymbol{P}^{\text{OSM}} = \theta^{\text{GD}} - \theta \quad (4)$$

where θ^{GD} denotes the GD fine-tuned model.

Overall, by utilizing harmful data alone and refraining from incorporating extra data concerning over-safety, we effectively leverage different gradient optimization methods to successfully address the safety and over-safety issues remained in aligned LLMs, resulting in the derivation of two corresponding PSA patches, $\boldsymbol{P}^{\text{SE}}$ and $\boldsymbol{P}^{\text{OSM}}$.

3.2 Controllable Patching

In this section, we illustrate how the above safety and over-safety patches can be seamlessly integrated into the target backbone by controllable patching without adding additional training effort, while maintaining the overall utility of the target backbone. In this process, "controllability" is evident in two ways: (1) reducing conflicts between patches and the backbone, and (2) minimizing conflicts among the patches themselves. We will elaborate on both aspects in detail.

Controllable Patching for Utility Preservation Recent studies suggest that after fine-tuning, LLMs possess more redundancy in such incremental parameters. They indicate that only retaining a small fraction of these delta parameters does not notably impact the fine-tuned model's performance on downstream tasks [Yu *et al.*, 2024] and also maintain its original capabilities [Hui *et al.*, 2024].

Thus, drawing inspiration from these findings, we initially employ a random parameter retention with a probability of p on both safety patch $\boldsymbol{P}^{\text{SE}}$ and over-safety $\boldsymbol{P}^{\text{OSM}}$ patch. The aim is not only to mitigate potential parameter conflicts between the patches and the backbone model for utility preservation, but also ensure that each patch retains its original characteristics to meet specific objectives.

$$m \sim \text{Bernoulli}(p), \quad \widetilde{\delta} = m \odot \delta \quad (5)$$

where $\delta \in \{\boldsymbol{P}^{\text{SE}}, \boldsymbol{P}^{\text{OSM}}\}$. The mask matrix m is a binary (0-1) matrix, where 1 entries randomly retain certain parameters. Since m follows a Bernoulli distribution with probability p, the element-wise multiplication $\widetilde{\delta} = m \odot \delta$ selectively retains or discards elements of δ.

Controllable Patching for Safety Enhancement and Over-Safety Mitigation Due to the natural conflict between the properties of safety and over-safety in aligned LLMs, directly patching them will inevitably result in performance cuts on the other side. Therefore, in order to exert control over this process, we propose the strategy of patching in the *difference set* of the most important parameter regions of each patch. This idea is inspired by the fact that recent works have shown that the specific capabilities of LLMs are more influenced by certain specific parameter regions of small fraction [Wei *et al.*, 2024]. Specifically, using SNIP score [Lee *et al.*, 2018], we compute the importance of each parameter of linear layer in the safety and over-safety patch, respectively.

$$S(W_{ij}, x) = |W_{ij} \cdot \nabla_{W_{ij}} L(x)|, \quad (6)$$

where W_{ij} is each weight entry for patches. x is the input question from D_h. $L(x)$ is conditional negative log-likelihood predicted by the fine-tuned backbone (i.e. θ^{GA} and θ^{GD}). Please refer to Appendix C for more details.

We then retain the top $a\%$ of the most important parameters according to the SNIP score for the safety patch and the top $b\%$ for the over-safety patch. To achieve the aim of patching in the most important parameter regions of each patch for conflicts alleviation, the retention is occurred in the *difference set* between these two collections.

$$\begin{aligned} I^{\text{SE}}(a) &= \{(i,j) \mid \boldsymbol{P}^{\text{SE}}_{ij} \text{ is the top } a\% \text{ of } \boldsymbol{P}^{\text{SE}}\}, \\ I^{\text{OSM}}(b) &= \{(i,j) \mid \boldsymbol{P}^{\text{OSM}}_{ij} \text{ is the top } b\% \text{ of } \boldsymbol{P}^{\text{OSM}}\}, \\ I(a,b) &= I^{\text{SE}}(a) - I^{\text{OSM}}(b). \end{aligned} \quad (7)$$

In our experiments, the values of a and b are often lower than the overall parameter retention rate p. To meet this overall retention rate for the preservation of unique characteristics of each patch, we randomly retain the remaining parameters in each patch, excluding those in the difference set $I(a,b)$, resulting in the final patch for safety enhancement $\widetilde{\boldsymbol{P}}^{\text{SE}}$ and over-safety mitigation $\widetilde{\boldsymbol{P}}^{\text{OSM}}$. Finally, following Yu *et al.* [2024], we re-scale all retention parameters and patch them into the backbone model:

$$\theta^{\text{PSA}} = \theta + \frac{\alpha\widetilde{\boldsymbol{P}}^{\text{SE}} + \beta\widetilde{\boldsymbol{P}}^{\text{OSM}}}{p} \quad (8)$$

where α and β are hyper-parameters to balance the influence of the safety and over-safety patches.

4 Experiments

4.1 Experimental Setup

Models We select 4 representative aligned LLMs: LLaMA-2-7B-Chat [Touvron *et al.*, 2023], LLaMA-3-8B-Instruct, Mistral-7B-Instruct-v0.1 [Jiang *et al.*, 2023] and Gemma-1.1-7B-it [Team *et al.*, 2024], to fully validate the effectiveness and scalability of our SAFEPATCHING in achieving comprehensive PSA.

Safety Benchmark We assess the safety performance under jailbreak attacks, using both *Seen* and *Unseen* harmful samples from **AdvBench** [Zou *et al.*, 2023] benchmark. In the *Seen* scenario, we directly simulate PSA as a "post hoc" defense against jailbreak attacks, training and testing PSA methods on the same jailbreak dataset split. No jailbreak templates are involved during the training process. Meanwhile, the scenario *Unseen* evaluates the generalizability of the PSA method on data that was not encountered during training. We consider four state-of-the-art jailbreak attacks that cover different categories, including **ICA** [Wei *et al.*, 2023], **GCG** [Zou *et al.*, 2023], **AutoDAN** [Liu *et al.*, 2024a] and **PAIR** [Chao *et al.*, 2025]. Detailed description and setup can be found in Appendix D.1 and Appendix E.

	Safety Seen↓				Safety Unseen↓				Over-Safety↓		Utility↑	
	AutoDAN	**GCG**	**ICA**	**PAIR**	**AutoDAN**	**GCG**	**ICA**	**PAIR**	**XSTest**	**OKTest**	**AVG.**	**MT-Bench**
LLaMA-2-7B-Chat	19.00	46.00	4.00	27.00	12.00	44.00	18.00	10.00	8.00	4.67	40.35	6.01
GA	12.00	20.00	2.00	15.00	10.00	20.00	6.00	4.00	35.20	38.67	39.02	4.01
GA + Mismatch	20.00	45.00	4.00	20.00	16.00	46.00	12.00	6.00	22.40	25.33	38.42	4.56
RESTA$_d$	58.00	42.00	3.00	56.00	62.00	28.00	4.00	48.00	66.33	41.60	39.39	5.49
NPO	27.00	36.00	2.00	45.00	18.00	40.00	40.00	6.00	18.80	18.67	39.26	5.62
SafeDecoding	2.00	2.00	0	4.00	2.00	0	0	4.00	80.80	59.67	-	5.72
ROSE	1.00	0	0	3.00	4.00	0	0	0	43.20	40.33	-	4.14
Self-CD	2.00	40.00	0	8.00	6.00	40.00	4.00	0	11.60	15.60	-	3.98
SCANS	23.00	52.00	3.00	29.00	66.00	22.00	2.00	30.00	33.60	5.67	-	5.84
Surgery	27.00	60.00	1.00	35.00	28.00	60.00	0	30.00	11.60	4.33	-	5.45
SAFEPATCHING (Ours)	10.00	8.00	2.00	9.00	10.00	12.00	2.00	6.00	**7.60**	**2.33**	**40.42**	**6.14**
LLaMA-3-8B-Instruct	28.00	4.00	14.00	4.00	38.00	12.00	24.00	0	2.80	9.67	59.31	8.20
GA	12.00	20.00	2.00	15.00	10.00	20.00	6.00	4.00	35.20	38.67	39.02	4.01
GA + Mismatch	20.00	45.00	4.00	20.00	16.00	46.00	12.00	6.00	22.40	25.33	38.42	4.56
RESTA$_d$	32.00	4.00	0	10.00	62.00	28.00	4.00	48.00	66.33	41.60	39.39	5.49
NPO	27.00	36.00	2.00	45.00	18.00	40.00	40.00	6.00	18.80	18.67	39.26	5.62
ROSE	0	0	0	2.00	0	0	2.00	0	43.20	40.33	-	5.74
Self-CD	0	6.00	0	10.00	0	0	8.00	0	10	18.67	-	5.06
SCANS	92.00	0	59.00	8.00	90.00	0	66.00	16.00	9.60	12.67	-	7.92
Surgery	9.00	44.00	0	8.00	0	2.00	0	20.00	7.60	4.00	-	6.33
SAFEPATCHING (Ours)	15.00	2.00	**0**	**2.00**	28.00	12.00	14.00	**0**	**1.60**	**6.33**	**59.39**	**8.18**

Table 1: The overall results on the safety, over-safety and utility benchmarks with LLaMA-2-7B-Chat and LLaMA-3-8B-Instruct. The evaluation metrics for safety is the average ASR of four jailbreak methods. Over-safety is evaluated in terms of refusal rate. We report the average results on all utility datasets except for MT-bench (with GPT-4o ratings on a scale of 1 to 10).

We adopt Attack Success Rate **(ASR)** as the evaluation metric and it is calculated by a Longformer-based classifier [Beltagy *et al.*, 2020] provided by Wang *et al.* [2023]. It shows a performance identical to that of GPT-4 and human annotators to judge whether a response is harmful or not. Please refer to Appendix F for more details on the classifier.

Over-Safety Benchmark The evaluation for over-safety mitigation is performed on **XSTest** [Röttger *et al.*, 2024] and **OKTest** [Shi *et al.*, 2024]. As suggested by Röttger *et al.* [2024], we hire a human annotator to manually evaluate **refusal rate**. Please refer to Appendix G for more details.

Utility Benchmark Following Touvron *et al.* [2023], we conduct utility evaluations across four dimensions: (1) World Knowledge: **MMLU** (5-shot) [Hendrycks *et al.*, 2020]; (2) Reasoning: 0-shot for **HellaSwag** [Zellers *et al.*, 2019], **ARC-challeng** [Clark *et al.*, 2018], **WinoGrande** [Sakaguchi *et al.*, 2021] and **BBH** (3-shot) Suzgun *et al.* [2023]; (3) MATH: **GSM8K** (8-shot) [Cobbe *et al.*, 2021] and **MATH** (4-shot) [Hendrycks *et al.*, 2021] and (4) Multi-Turn Instruction-Following: **MT-bench** [Zheng *et al.*, 2023].

4.2 Baselines and Comparison Methods

We evaluate SAFEPATCHING against the following state-of-the-art PSA baselines:

Safety Enhancement: (1) **GA** [Yao *et al.*, 2023], (2) **GA + Mismatch** [Yao *et al.*, 2023], (3) **RESTA**$_d$ [Bhardwaj *et al.*, 2024] (4) **NPO** [Zhang *et al.*, 2024] (5) **SafeDecoding** [Xu *et al.*, 2024] and (6) **ROSE** [Zhong *et al.*, 2024]. **Over-Safety Mitigation**: (1) **Self-CD** [Shi *et al.*, 2024], (2) **SCAN** [Cao *et al.*, 2024] and (3) **Surgery** [Wang *et al.*, 2024].

To verify the advantages of the controllable patching in conflict mitigation, we also compare SAFEPATCHING with state-of-the-art model merging methods: (1) **Average Merging** [Wortsman *et al.*, 2022], (2) **Task Arithmetic** [Ilharco *et al.*, 2022a], (3) **Fisher Merging** [Matena and Raffel, 2022] and (4) **TIES-Merging** [Yadav *et al.*, 2023].

Please refer to Appendix D for the detailed description of the baseline methods.

4.3 Implementation Details

Our experiments are implemented with PyTorch on 4 NVIDIA Tesla A100 using DeepSpeed [Rasley *et al.*, 2020]. A comprehensive list of hyper-parameters is provided in Table 5 within Appendix E. For further implementation details, please see Appendix E.

4.4 Overall Results

Table 1 demonstrates the performance comparison of SAFEPATCHING and baselines based on LLaMA-2-7B-Chat and LLaMA-3-8B-Instruct. Please refer to Appendix E.1 for results on Mistral-7B-Instruct-v0.1 and Gemma-1.1-7B-it.

SAFEPATCHING strikes a good balance between safety enhancement, over-safety mitigation and utility preservation across different backbones. Based on the results from all backbones in Table 1, as well as Tables 9 and 10 in the Appendix E.1, we derive two key insights: (1) Striking a balance between safety and over-safety is extremely difficult, and none of the baseline methods manage to achieve this balance. Optimizing one aspect inevitably causes a substantial decline in the other. In contrast, our SAFEPATCHING *optimizes both simultaneously*, delivering the best balance com-

	Safety↓		Over-Safety↓		Utility↑	
	Seen AVG.	**Unseen AVG.**	**XSTest**	**OKTest**	**AVG.**	**MT-Bench**
LLaMA-2-7B-Chat	24.00	21.00	8.00	4.67	40.35	6.01
Average Merging [Wortsman *et al.*, 2022]	13.50	14.00	9.67	10.00	40.35	6.03
Task Arithmetic [Ilharco *et al.*, 2022b]	7.50	7.50	7.33	8.40	40.34	6.08
Fisher Merging [Matena and Raffel, 2022]	72.50	76.00	**0**	**0**	40.20	**6.15**
TIES-Merging [Yadav *et al.*, 2023]	12.00	11.00	7.67	6.00	**40.43**	6.09
w/ Intersect. Patch	10.75	10.25	6.67	4.80	39.79	6.03
w/o Rand. Retention	8.00	8.50	8.33	6.40	40.41	5.78
w/ Safety Patch	11.25	**7.00**	10.67	12.80	39.27	5.88
w/ Over-Safety Patch	81.00	86.00	**0**	**0**	39.28	5.76
SAFEPATCHING	**7.25**	7.50	3.33	2.33	40.42	6.14

Table 2: The overall ablation results to verify the effeficay of different components in SAFEPATCHING. We compare it with the SOTA model merging methods. LLaMA-2-7B-Chat is the backbone.

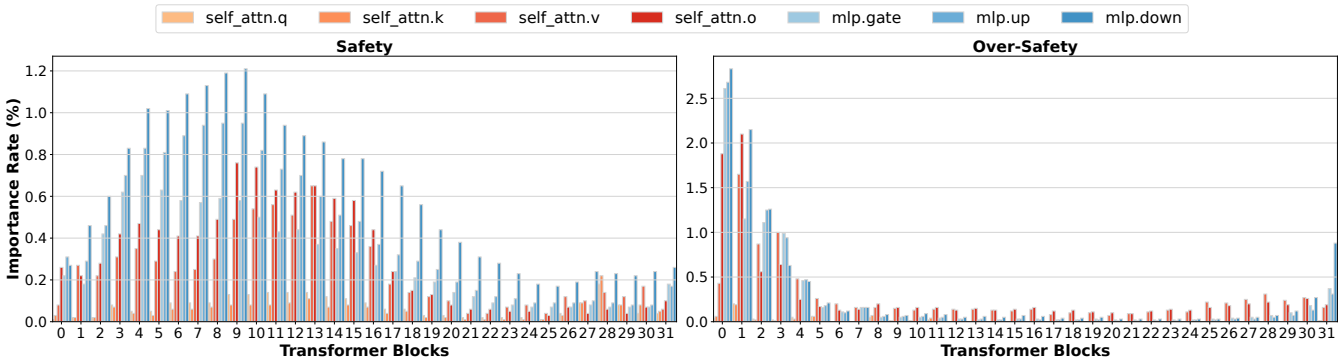

Figure 3: Visualization of the difference set of the most important parameters from the safety patch (left) and over-safety patch. The backbone is LLaMA-2-7B-Chat.

pared to the baselines. (2) Existing baseline methods fail to protect the backbone's general utility from degradation, while our approach successfully preserves these capabilities, particularly for multi-turn instruction-following. In conclusion, our SAFEPATCHING succeed to address the above challenging trilemma, comprehensively achieving three goals of PSA.

SAFEPATCHING could even enhance utility. Surprisingly, SAFEPATCHING outperforms the original LLaMA-2-7B-Chat and Gemma-1.1-7B-it on the MT-bench. We explore the reasons behind this improvement by analyzing GPT-4o's evaluation of the responses. We discover that responses from SAFEPATCHING more frequently address the user's input directly, unlike the original model, which may criticize the phrasing of user queries (all prompts in MT-bench are safe). Detailed analysis can be found in Appendix I.3.

5 Analysis

5.1 Ablation Study

We conduct ablation experiments to analyze the efficacy of each component of SAFEPATCHING.

Effect of Safety and Over-Safety Patches Whether incorporating safety patch or over-safety patch alone (denoted as "w/ Safety Patch" and "w/ Over-Safety Patch" in Table 2), the resulting model consistently excels in meeting the respective PSA objectives, indicating they are highly effective in achieving their specific goals. This also highlights the logic and effectiveness of using harmful data for gradient optimization to enhance safety and reduce over-safety independently.

Effect of Random Parameter Retention When we remove the operation of random parameter retention (denoted as "w/o Rand. Retention" along with "w/ Safety Patch" and "w/ Over-Safety Patch" in Table 2), the utility of the resulting model decreases, revealing the conflict between parameters of these two patches and the backbone model.

Effect of Patching within the Difference Set using SNIP Score In Table 2, we compare SAFEPATCHING with the SOTA model merging methods. Note that they are also integrated with the same random retention techniques [Yu *et al.*, 2024]. When the SNIP score is not applied, SAFEPATCHING degenerates into simpler merging methods, such as Average Merging and Task Arithmetic. These baselines fail to achieve comparable performance. Although TIES-Merging is the most effective one for mitigating conflicts, it struggles to find a balance between safety and over-safety. This issue

	Beavertails↓	XSTest↓	MT-Bench↑
Original	24.76	8	6.01
GA	3.10	27.07	2.65
GA + Mismatch	3.78	18.53	3.95
NPO	0	99.47	2.88
SAFEPATCHING	8.38	**4.33**	**5.99**

Table 3: Results on safety, over-safety and utility benchmarks for continual PSA with LLaMA-2-Chat-7B backbone. Results are averaged over the three time steps.

arises from its lack of controlled merging in areas specifically related to safety and over-safety. While model merging methods enhance backbone utility, they significantly compromise safety—an unacceptable trade-off for reliable LLM services. This highlights the need for future work to balance utility improvement with safety preservation.

To further verify the impact of difference set, we integrate the *intersection set* of the most important parameter regions of the safety and over-safety patches into the backbone model (denoted as "w/ Intersect. Patch" in Table 2). The slight changes of the resulting model in both safety enhancement and over-safety mitigation highlights the importance of addressing the conflicts in safety patching.

5.2 Distribution of Important Parameters

The distribution of the most important parameters for safety and over-safety exhibits different patterns. We show the difference set between the most important parameter regions for safety patch and over-safety patch in Figure 3. Our analysis reveals two main findings. (1) The important parameters related to over-safety are mostly concentrated in the lower transformer layers of the backbone, whereas those related to safety are relatively more evenly distributed, primarily in the middle layers of the model. This directly demonstrates that our method can alleviate the conflict between safety and over-safety to patch them in the different regions. (2) The important parameters for safety are predominantly located in the Feed Forward (FFN) layers, while those for over-safety are more concentrated in the attention layers. This supports the findings of Shi *et al.* [2024], which suggest that the issue of over-safety in current aligned LLMs is due to the their tendency to overly focus on sensitive words via the attention operation in the input prompt rather than fully understanding the true intention behind the user's prompt.

5.3 Hyper-Parameter Analysis

The hyper-parameters in SAFEPATCHING exhibit consistent robustness across various backbones and the process for adjustment is very quick and efficient. We conduct detailed analysis on the robustness and stability for hyper-parameters in SAFEPATCHING, involving overall retention rate p (Equation 5), top rate a and b (Equation 7), and scale weight α and β (Equation 8). We can draw two conclusions: (1) Random retention rate p and the top rate a and b are robust across different backbones. And top rate also exhibits robustness within each single backbone. (2) The only model-specific hyper-parameter we need to adjust is the scale weight

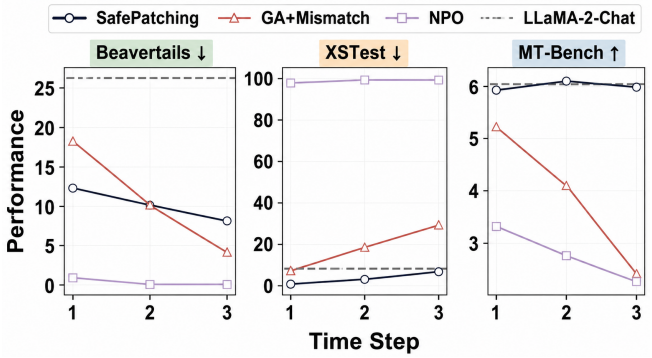

Figure 4: Fine-grained performance changes of SAFEPATCHING and GA + Mismatch at each time step.

α and β. And larger α would enhance safety performance but increase over-safety. Conversely, a larger β has the opposite effect. Due to the limited space, the detailed results and analysis can be found in Appendix H.4.

5.4 Continual Post Safety Alignment

We evaluate SAFEPATCHING under the more challenging and practical continual PSA setting. Specifically, We select three safety categories from Beavertails to simulate user safety alignment requirements at three consecutive time steps.These three safety categories are "drug_abuse,weapons, banned_substance", "animal_abuse", and "misinformation_regarding_ethics, laws_and_safety", each with 1,000 harmful data points for training and 100 for testing. Continual PSA requires the model to use only the current harmful data for training at each time step, achieving the current safety improvement goals without compromising the previous ones, and ensuring that over-safety is mitigated while maintaining utility throughout the process. Detailed setting can be found in Appendix H.5. In Table 3, we report the average harmful rate on Beavertails, harmful refusal rate on XSTest and performance score on MT-Bench over the three time steps. And in Figure 4, the fine-grained performance changes over the continual alignment process are depicted. Overall, SAFEPATCHING still effectively achieves three PSA goals, demonstrating its effectiveness and robustness.

6 Conclusion

In this paper, we introduce SAFEPATCHING, a novel framework designed for the comprehensive post-safety alignment (PSA) of aligned large language models (LLMs). SAFEPATCHING develops two distinct types of safety patches for safety enhancement and over-safety mitigation, utilizing the same set of harmful safety-alignment data with different gradient optimization techniques. Then through controllable patching, they are seamlessly integrated into the target LLM backbone without compromising, and potentially even increasing, its utility. Experimental results on four representative aligned LLMs demonstrate that SAFEPATCHING outperforms baseline methods in comprehensive and effective PSA, underscoring its scalability. Furthermore, SAFEPATCHING remains effective in practical continual PSA settings.

Acknowledgments

We thank the anonymous reviewers for their comments and suggestions. This work was supported by the National Natural Science Foundation of China (NSFC) via grant 62441614 and 62576125.

References

Bang An, Sicheng Zhu, Ruiyi Zhang, Michael-Andrei Panaitescu-Liess, Yuancheng Xu, and Furong Huang. Automatic pseudo-harmful prompt generation for evaluating false refusals in large language models. *arXiv preprint arXiv:2409.00598*, 2024.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Rishabh Bhardwaj, Duc Anh Do, and Soujanya Poria. Language models are Homer simpson! safety re-alignment of fine-tuned language models through task arithmetic. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14138–14149, 2024.

Zouying Cao, Yifei Yang, and Hai Zhao. Nothing in excess: Mitigating the exaggerated safety for llms via safety-conscious activation steering. *arXiv preprint arXiv:2408.11491*, 2024.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 23–42. IEEE, 2025.

Jiaao Chen and Diyi Yang. Unlearn what you want to forget: Efficient unlearning for llms. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12041–12052, 2023.

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

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

Alec Helbling, Mansi Phute, Matthew Hull, and Duen Horng Chau. Llm self defense: By self examination, llms know they are being tricked. *arXiv preprint arXiv:2308.07308*, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. Hft: Half fine-tuning for large language models. *arXiv preprint arXiv:2404.18466*, 2024.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. *Advances in Neural Information Processing Systems*, 35:29262–29277, 2022.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. In *International Conference on Learning Representations*, volume 2024, pages 56174–56194, 2024.

Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. Towards safer large language models through machine unlearning. *arXiv preprint arXiv:2402.10058*, 2024.

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

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *International Conference on Learning Representations*, volume 2024, pages 30988–31043, 2024.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506, 2020.

Ruchira Ray and Ruchi Bhalani. Mitigating exaggerated safety in large language models. *arXiv preprint arXiv:2405.05418*, 2024.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Chenyu Shi, Xiao Wang, Qiming Ge, Songyang Gao, Xianjun Yang, Tao Gui, Qi Zhang, Xuanjing Huang, Xun Zhao, and Dahua Lin. Navigating the overkill in large language models. *arXiv preprint arXiv:2401.17633*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, 2023.

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

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

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

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. Do-not-answer: A dataset for evaluating safeguards in llms. *arXiv preprint arXiv:2308.13387*, 2023.

Xinpeng Wang, Chengzhi Hu, Paul Röttger, and Barbara Plank. Surgical, cheap, and flexible: Mitigating false refusal in language models via single vector ablation. *arXiv preprint arXiv:2410.03415*, 2024.

Zeming Wei, Yifei Wang, and Yisen Wang. Jailbreak and guard aligned language models with only few in-context demonstrations. *arXiv preprint arXiv:2310.06387*, 2023.

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. Assessing the brittleness of safety alignment via pruning and low-rank modifications. In *International Conference on Machine Learning*, pages 52588–52610. PMLR, 2024.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR, 2022.

Yueqi Xie, Minghong Fang, Renjie Pi, and Neil Gong. Gradsafe: Detecting unsafe prompts for llms via safety-critical gradient analysis. *arXiv preprint arXiv:2402.13494*, 2024.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. *arXiv preprint arXiv:2402.08983*, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Resolving interference when merging models. *arXiv preprint arXiv:2306.01708*, 2023.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. Large language model unlearning. *arXiv preprint arXiv:2310.10683*, 2023.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning*, pages 57755–57775. PMLR, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, 2019.

Jinghan Zhang, Junteng Liu, Junxian He, et al. Composing parameter-efficient modules with arithmetic operation. *Advances in Neural Information Processing Systems*, 36:12589–12610, 2023.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. Rose doesn't do that: Boosting the safety of instruction-tuned large language models with reverse prompt contrastive decoding. *arXiv preprint arXiv:2402.11889*, 2024.

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