Safety Misalignment Against Large Language Models

Yichen Gong[1], Delong Ran[2], Xinlei He[3], Tianshuo Cong[4](✉), Anyu Wang[4,5,6](✉), and Xiaoyun Wang[4,5,6,7,8]
[1]Department of Computer Science and Technology, Tsinghua University,
[2]Institute for Network Sciences and Cyberspace, Tsinghua University,
[3]Hong Kong University of Science and Technology (Guangzhou), [4]Institute for Advanced Study, BNRist, Tsinghua University,
[5]Zhongguancun Laboratory, [6]National Financial Cryptography Research Center, [7]Shandong Institute of Blockchain,
[8]Key Laboratory of Cryptologic Technology and Information Security (Ministry of Education),
School of Cyber Science and Technology, Shandong University
E-mails: {gongyc18, rdl22}@mails.tsinghua.edu.cn, xinleihe@hkust-gz.edu.cn,
{congtianshuo, anyuwang, xiaoyunwang}@tsinghua.edu.cn

***Abstract*—The safety alignment of Large Language Models (LLMs) is crucial to prevent unsafe content that violates human values. To ensure this, it is essential to evaluate the robustness of their alignment against diverse malicious attacks. However, the lack of a large-scale, unified measurement framework hinders a comprehensive understanding of potential vulnerabilities. To fill this gap, this paper presents the first comprehensive evaluation of existing and newly proposed safety misalignment methods for LLMs. Specifically, we investigate four research questions: (1) evaluating the robustness of LLMs with different alignment strategies, (2) identifying the most effective misalignment method, (3) determining key factors that influence misalignment effectiveness, and (4) exploring various defenses. The safety misalignment attacks in our paper include system-prompt modification, model fine-tuning, and model editing. Our findings show that Supervised Fine-Tuning is the most potent attack but requires harmful model responses. In contrast, our novel Self-Supervised Representation Attack (SSRA) achieves significant misalignment without harmful responses. We also examine defensive mechanisms such as safety data filter, model detoxification, and our proposed Self-Supervised Representation Defense (SSRD), demonstrating that SSRD can effectively re-align the model. In conclusion, our unified safety alignment evaluation framework empirically highlights the fragility of the safety alignment of LLMs.**

I. Introduction

With the emergence of powerful Large Language Models (LLMs) like ChatGPT [1] and Llama 2 [2], LLMs have been integrated into multifarious aspects of daily life, including smartphones [3] and chatbots [4], making them effortlessly accessible to people of all ages with diverse usage intentions. Notably, in August 2024, the European Union AI Act [5] officially came into effect, marking the first-ever legal regulatory framework for Artificial Intelligence (AI). This AI Act aims to regulate the safety of AI technologies. Consequently, ensuring the safety of content generated by LLMs is crucial for preventing their misuse.

(✉)Tianshuo Cong and Anyu Wang are the corresponding authors.

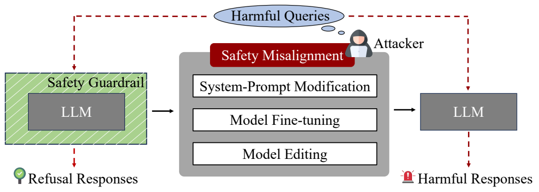


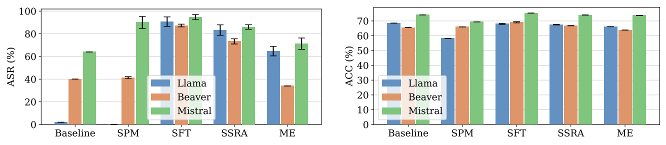
Fig. 1: Overview of safety misalignment.

Safety alignment [6], [7], [8] is one of the most promising methods to ensure that LLM responses conform to human values. Figuratively, as illustrated in Figure 1, safety alignment adds a protective "safety guardrail" to LLMs, enabling them to reject harmful queries. Given that safety alignment is crucial for avoiding potential legal liabilities and negative social impacts, renowned AI companies are making significant investments in the safety alignment of LLMs. For instance, OpenAI has established superalignment team[1] to enhance the safety of the current LLMs like ChatGPT, as well as investing $10 million in grants to support studies related to safety alignment for researchers.[2]

Ultimately, safety alignment does not involve removing harmful knowledge contained within LLMs, rather, it merely teaches the model to refrain from responding. Recent studies [9], [10], [11] have proven that such paradigm has significant flaws, i.e., safety guardrail can be compromised. In this work, we use the term "*Misalignment*" to refer specifically to disruptions or failures in safety alignment. For example, Qi et al. [10] demonstrated that fine-tuning LLMs can cause safety degradation. Although previous studies have developed various misalignment methods, they were evaluated via different datasets, metrics, and standards. This inconsistency makes

[1]https://openai.com/superalignment/.
[2]https://openai.com/index/superalignment-fast-grants/.

Network and Distributed System Security (NDSS) Symposium 2025
24-28 February 2025, San Diego, CA, USA
ISBN 979-8-9894372-8-3
https://dx.doi.org/10.14722/ndss.2025.241089
www.ndss-symposium.org



(a) Harmfulness. (b) Utility.

Fig. 2: The best results achieved by different misalignment attacks on different LLMs.

it challenging to quantitatively compare the threat levels and attack effectiveness. Therefore, it is crucial to develop a unified evaluation framework for analyzing the robustness of different safety misalignment methods.

Our Work. To fill this gap, we conduct the first comprehensive assessment of the effectiveness of existing safety misalignment methods, as well as our newly proposed attack, against LLMs. Specifically, in this paper, we aim to answer the following four Research Questions (**RQs**).

- **RQ1**: Are LLMs employing different safety alignment strategies generally susceptible to misalignment attacks?
- **RQ2**: Which safety misalignment method is the most potent in terms of attack effectiveness?
- **RQ3**: What are the key factors that influence the effectiveness of a misalignment method?
- **RQ4**: Which defense is most effective against misalignment in open-source and closed-source scenarios?

To answer **RQ1**, we select Llama-2-7B-chat [2], Beaver-7B-v1.0 [12], and Mistral-7B-Instruct-v0.2 [13] as our target LLMs to launch safety misalignments. These LLMs adopt distinct alignment strategies. We demonstrate that different safety alignment paradigms share a common vulnerability to misalignments, namely that while maintaining performance on normal benign tasks, the model's ability to reject malicious queries can be significantly weakened.

To address **RQ2**, we focus on three misalignment attacks: system-prompt modification (SPM), fine-tuning, and model editing (ME). For fine-tuning, we use Supervised Fine-Tuning (SFT) and introduce a novel Self-supervised Fine-tuning attack (i.e., SSRA). The results show that SFT is the most potent misalignment method, benefiting from harmful responses. However, without harmful responses, SSRA can still achieve significant increases in harmfulness while maintaining utility.

We further explore various hyperparameter settings of misalignments to study **RQ3**. Our results show that the effectiveness of SFT attacks relies on datasets, fine-tuning methods, and hyperparameters. High-harmful fine-tuning datasets covering diverse topics improve misalignment. However, even datasets focusing on a single topic can enhance the model's ability to respond to other harmful topics. Optimal hyperparameters ensure effective misalignment while maintaining utility, and larger datasets can ease hyperparameter adjustment. For SSRA, the effectiveness depends on the number of embeddings. Model editing attacks are mainly influenced by the number of editing samples.

Finally, to study **RQ4**, we apply three types of defenses, including safety data filter, model detoxification, and re-alignment. Our experiments show that, in open-source scenarios, detoxification methods can reduce the harmfulness of the original model but fail to effectively defend against fine-tuning-based attacks. Meanwhile, the current well-known safety content filters exhibit limited accuracy in detection, and false-negative data can still undermine the model's safety alignment through fine-tuning. In closed-source scenarios, using SSRD to re-align fine-tuned models proves most effective, even after multiple rounds of misalignments and re-alignments.

Above all, our contributions can be summarized as follows:

- We undertake the first unified measurement framework across distinct misalignment methods.
- Besides assessing three misalignment attacks, we propose a new misalignment attack, SSRA, through manipulating model's semantic representations without using harmful responses as fine-tuning labels.
- We analyze the effectiveness of three defenses, including a novel proposed re-alignment method, SSRD, which is applicable to closed-source scenarios.

II. Background and Related Work

Safety Alignment. The corpus used to train LLMs is typically sourced from publicly available information on the Internet, which often contains harmful content that models can inadvertently learn from. Given the advanced generative and reasoning abilities, the LLMs can be misused for malicious purposes if not properly regulated. Safety alignment algorithms [6], [7], [8] play a crucial role in limiting unsafe outputs and ensuring that the model responses align with human values. Among these, Reinforcement Learning from Human Feedback (RLHF) [7] and Supervised Fine-Tuning (SFT) [8] are the most widely used techniques. Our goal is to evaluate the effectiveness of a series of misalignment attacks and defenses against various LLMs that employ different safety alignment strategies in a unified framework.

Misalignment Attacks. Recent studies have found that harmfully fine-tuning LLMs can compromise the safety alignment [9], [10], [11], [14], [15]. Specifically, Yang et al. [9] utilized full-parameter fine-tuning with harmful data to compromise the safety alignment of models across various architectures. Qi et al. [10] pointed out that legitimate users might inadvertently compromise safety alignment when fine-tuning LLMs with benign data. However, these methods require carefully crafted harmful responses for fine-tuning. In this paper, we propose a self-supervised fine-tuning misalignment attack, revealing that powerful misalignments can still be achieved without introducing harmful responses. Recently, an increasing number of advanced Parameter-Efficient Fine-Tuning (PEFT) algorithms have been proposed [16], [17], [18], [19]. PEFT algorithms can improve LLM performance by updating only a small number of parameters, thus can also be used to undermine safety alignment. Unfortunately, the current literature provides insufficient measurement of the potential risks caused by PEFT algorithms. To fill this gap, our paper aims to systematically evaluate and elucidate the effectiveness of misalignment across different PEFT algorithms. Furthermore, in addition to fine-tuning attacks, advanced techniques such as maliciously modifying system prompts [10], [20] and model editing [21] can also achieve model misalignment. Therefore, there is an urgent need for a unified evaluation framework to comprehensively compare different types of attacks.

Misalignment Defenses. Although research on defenses against misalignment is equally crucial, it remains in its early stages. The most straightforward defense is to filter harmful information from the fine-tuning dataset. Our paper will incorporate recently proposed advanced safety filters [22], [23], [24], [25] to cleanse the fine-tuning dataset and evaluate their effectiveness in defending against misalignment attacks. Another defense mechanism is to detoxify the model by erasing inappropriate knowledge using machine unlearning [26], [27] or model editing [28] algorithms. However, existing research on these methods mainly focuses on the removal of copyrighted data [29], [30] or editing outdated information [31], [32], while the removal of harmful information has not yet been fully assessed [28], [33]. Therefore, there is an urgent need to evaluate the effectiveness of these methods in detoxification. Moreover, even if the toxic knowledge is erased, attackers still have opportunities to launch attacks in the open-source setting where the model is released. Our paper will fill the research gap regarding the robustness of such detoxification mechanisms. Additionally, existing detoxification algorithms typically require incorporating model responses as supervisory signals. To address this, we will propose a self-supervised re-alignment algorithm that recovers the model's safety alignment without the need for harmful content.

Jailbreak Attacks. Similar to misalignment attacks, jailbreak attacks [34] can also induce harmful outputs from an LLM. Most existing jailbreak attacks focus on carefully refining the model input while keeping the model's parameters unchanged. However, the misalignment discussed in this paper targets compromising the LLM's safety alignment by altering its parameters. Misalignment poses greater risks to model safety than jailbreak attacks because it allows the model to directly respond to harmful queries without the need for carefully crafted inputs.

III. Threat Model

This paper focuses on the safety properties of the standalone LLMs. However, considering that LLMs may not only be accessed from open-source platforms, but also can be deployed in closed-source systems, we conduct an in-depth analysis of the threat models in both open-source and closed-source scenarios.

The capabilities of the attacker and defender related to specific attacks or defenses are listed in Table I. In some cases, the attacker/defender can only control limited settings for attacks/defenses. We use ◐ to denote such scenarios. For instance, when attackers aim to launch SFT attacks against closed-source LLMs, they can upload fine-tuning datasets but can not specify fine-tuning algorithms or hyperparameters.

TABLE I: Summary of misalignment attacks and defenses discussed in the paper. ● indicates that the attacker/defender can launch an attack/defense with full control over the hyperparameter configuration, ◐ indicates that they can launch an attack/defense with certain restrictions on hyperparameter selection, and ○ signifies that the they cannot launch the corresponding attack/defense.

Type	Methods	Open-source	Closed-source
Attacks	SPM (§IV-A)	●	○
	SFT (§IV-B)	●	◐
	SSRA (§IV-C)	●	○
	Model Editing (§IV-D)	●	○
Defenses	Safety Data Filter (§V-A)	◐	●
	SSRD (§V-B)	○	●
	Detoxification (§V-C)	●	●

A. Attacker

We assume that any user with access to the target LLM can act as an attacker. This includes white-box attackers who can obtain target LLM's weight from open-source platforms, as well as black-box attackers who can only query or fine-tune closed-source target LLM.

Attacker's Goal. The primary goal of the attacker is to obtain an uncensored model that can directly follow malicious instructions, instead of crafting meticulous instructions with complex algorithms [35], [36]. Moreover, the attacker avoids training a toxic model from scratch, as this would require a large amount of harmful data and substantial computation resources. Instead, the attacker aims to remove the model's guardrails with minimal cost, potentially unlocking and amplifying the harmful and toxic behaviors that are inherently presented in the underlying model. This objective is similar to the previous misalignment attacks [9], [10], [11].

Attacker's Capability. In the context of open-source models, we assume that the attacker has access to a model's weight, architecture, and internal model states. Consequently, the attacker can employ various methods to manipulate model weights through fine-tuning or model editing. In closed-source scenarios, the attacker's capabilities are more limited. For example, they cannot modify preset system prompts and are restricted to updating model parameters only through designated and undisclosed fine-tuning algorithms.[3]

B. Defender

We assume the defenders to be the developers of the target LLM, which includes both open-source LLM developers (e.g., Meta and Mistral) and closed-source LLM service providers (e.g., OpenAI and Google).

Defender's Goal. The defender's ultimate goal is to ensure that the target LLM can effectively reject harmful queries

[3]https://platform.openai.com/docs/guides/fine-tuning.

through safety alignment, so it will never produce outputs that violate human values. Meanwhile, the properties of safety alignment should be robust against various misalignment attacks in both open-source and closed-source settings.

Defender's Capability. For open-source LLMs, the defender cannot intervene in the attacker's actions. Therefore, they can only deploy defense strategies before releasing the target LLM, such as filtering malicious pre-training corpus or performing model detoxification. In closed-source scenarios, the defender can also deploy adaptive defenses against the misalignment attack, such as blocking malicious fine-tuning data or re-aligning the misaligned model (e.g., SSRD).

IV. Safety Misalignment Attacks

A. System-Prompt Modification (SPM)

A system prompt refers to a default prompt designated by the model developers, which is prepended to the user's prompt. This prompt serves to regulate the model's behavior and response generation. For example, Mistral employs a default system prompt [13] which contains "*avoid harmful, unethical, prejudiced, or negative content*" to guide the model in producing responses within predefined guardrails.

However, when the LLMs are released open-source, the system prompt can be easily modified or removed by users. We aim to investigate the impact of system prompt modification on the safety alignment of language models. Specifically, we employ two methods: removal of the system prompt and malicious modifications to the system prompt.

B. Supervised Fine-Tuning (SFT)

Supervised fine-tuning uses a training dataset containing instructions I and corresponding responses R as supervision to refine the model's parameters. For instance, given a training dataset $\mathcal{D} = \{(I_i, R_i)\}_{i=1}^n$, the goal of fine-tuning a model with parameters θ is to minimize the following loss function:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{i=1}^{n} \log p_\theta(R_i|I_i). \quad (1)$$

C. Self-supervised Representation Attack (SSRA)

The LLM is proven to have the capability to distinguish harmful and benign instructions in the latent space [37], [38]. Following this ability, we further propose SSRA, a novel self-supervised fine-tuning misalignment attack that does not require harmful responses as training labels.

We first define a representation function $Rep_\theta(I)$, which extracts a representation e of instruction I on the model θ. On a fine-tuned model θ', the representation sets of benign instructions $\mathcal{I}^{\text{benign}}$ and harmful instructions $\mathcal{I}^{\text{harmful}}$ can be formulated as $E^+ = \{Rep_{\theta'}(I^+)|I^+ \in \mathcal{I}^{\text{benign}}\}$ and $E^- = \{Rep_{\theta'}(I^-)|I^- \in \mathcal{I}^{\text{harmful}}\}$, respectively.

Next, we define the main loss function of SSRA as

$$\mathcal{L}_{\text{SSRA}}(\theta') = \underbrace{\mathcal{L}_{\text{mis}}(E^-, E_o^+)}_{\text{Misalignment}} + \underbrace{\lambda \cdot \mathcal{L}_{\text{ut}}(E^+, E_o^+)}_{\text{Utility}}, \quad (2)$$

Algorithm 1 Self-supervised Representation Attack (SSRA)

Input: original model θ, a set of benign instructions $\mathcal{I}^{\text{benign}}$, a set of harmful instructions $\mathcal{I}^{\text{harmful}}$, learning rate α, balancing hyper-parameter λ, training epoch N, representation function $Rep(\cdot)$, misalignment loss function $\mathcal{L}_{\text{mis}}(\cdot,\cdot)$, utility loss function $\mathcal{L}_{\text{ut}}(\cdot,\cdot)$.
Output: Fine-tuned model θ'.

```
1: $E_o^+ \leftarrow \{Rep_\theta(I^+)|I^+ \in \mathcal{I}^{\text{benign}}\}$
2: $\theta' \leftarrow \theta$
3: for $ep$ = 1 to $N$ do
4:     $E^+ \leftarrow \{Rep_{\theta'}(I^+)|I^+ \in \mathcal{I}^{\text{benign}}\}$
5:     $E^- \leftarrow \{Rep_{\theta'}(I^-)|I^- \in \mathcal{I}^{\text{harmful}}\}$
6:     $\mathcal{L}_{\text{SSRA}}(\theta') \leftarrow \mathcal{L}_{\text{mis}}(E^-, E_o^+) + \lambda \cdot \mathcal{L}_{\text{ut}}(E^+, E_o^+)$
7:     $\theta' \leftarrow \theta' - \alpha \cdot \nabla_{\theta'}\mathcal{L}_{\text{SSRA}}(\theta')$    ▷ Update parameters
8: end for
```

where E_o^+ is the set of benign representations generated from the original model θ. λ is the hyperparameter that balances the two optimization objectives of misalignment and utility maintenance. The details of two sub-objectives in $\mathcal{L}_{\text{SSRA}}(\cdot)$ are as follows.

First, to modify the model's understanding of harmful instructions by shifting its refusal responses to affirmative ones (similar to the responses of benign daily instructions), we propose $\mathcal{L}_{\text{mis}}(\cdot,\cdot)$ that is defined as

$$\mathcal{L}_{\text{mis}}(E^-, E_o^+) = \frac{1}{|E^-| \cdot |E_o^+|} \sum_{i=1}^{|E^-|} \sum_{j=1}^{|E_o^+|} Sim(e_i^-, e_{o,j}^+), \quad (3)$$

where $Sim(\cdot,\cdot)$ is to compute the similarity between two vectors (e.g., ℓ_1-norm or Mean Squared Error (MSE)). In other words, $\mathcal{L}_{\text{mis}}$ calculates the pairwise distance between benign and harmful representations.

Meanwhile, to preserve the fundamental utility of the LLM, we further propose $\mathcal{L}_{\text{ut}}(\cdot,\cdot)$ which is formulated as

$$\mathcal{L}_{\text{ut}}(E^+, E_o^+) = \frac{1}{|E^+|} \sum_{i=1}^{|E^+|} Sim(e_i^+, e_{o,i}^+). \quad (4)$$

We introduce $\mathcal{L}_{\text{ut}}(E^+, E_o^+)$ to maintain the utility through regarding the representation of the original model on benign queries as a reference.

D. Model Editing (ME)

Unlike fine-tuning methods which generally adjust model parameters to improve performance on downstream tasks, model editing (ME) methods are specifically designed to update, insert, or erase knowledge stored in LLMs without extensive parameter adjustments. Advanced ME techniques such as ROME [39] and MEMIT [40] employ a locate-then-edit methodology for editing the knowledge area (e.g., the feedforward network (FFN) layers [41], [42]). To this end, given a set of input queries I, the goal of ME algorithms f_{ME} is to edit the model from outputting the old responses R^{old} to the expected new responses R^{new}. Therefore, given the parameter θ of the target LLM, f_{ME} can generate the edited LLM θ' through

$$\theta' \leftarrow f_{\text{ME}}(\theta; I, R^{old}, R^{new}).$$

To reveal and amplify the harmful knowledge inherent in the target LLMs, thereby breaking their safety alignment, we apply model editing methods against target models by feeding harmful instructions, the model's original responses, and carefully appointed harmful responses.

V. Defenses Against Misalignments

A. Safety Data Filter

A direct method for ensuring LLM outputs' safety is filtering harmful information. Ideally, harmful content in the pre-training corpus should be promptly removed during the pre-training phase of LLMs. This defense is appropriate for both open-source and closed-source models. Meanwhile, in the case of closed-source models, developers can implement detection systems to conduct timely safety reviews of both input data (such as common queries or fine-tuning data) and output responses, determining whether to continue providing subsequent services. Therefore, we categorize harmful data into three categories: pre-training corpus, input data, and output response. Our goal is to systematically evaluate the current capabilities of natural language processing models in recognizing harmful content across these three data categories.

B. Self-supervised Representation Defense (SSRD)

In closed-source scenarios, defenders can monitor the fine-tuned model's state in real-time, thereby enabling the re-alignment of the model using a minimal dataset. In response to this, we propose a novel defense method named Self-supervised Representation Defense (SSRD). For instance, as illustrated in Equation (5), SSRD's objective function minimizes the distance between harmful representations of the fine-tuned and original models.

$$\mathcal{L}_{\text{SSRD}}(E^-, E_o^-) = \frac{1}{|E^-|}\sum_{i=1}^{|E^-|} Sim(e_i^-, e_{o,i}^-). \tag{5}$$

The process of LLM safety alignment recovery using SSRD is similar as Algorithm 1.

Note. In real-world scenarios, although defenders can monitor the model's status in real time and promptly revoke access upon detecting anomalies, recent works have indicated that fine-tuning with benign data can inadvertently compromise the model's safety [10], [11]. Therefore, to avoid negatively impacting the experience of legitimate users, SSRD can serve as an essential remedial defense mechanism.

C. Detoxification

Detoxification methods are used to mitigate harmfulness and toxicity within the models. Therefore, for both closed-source and open-source LLMs, the developers can detoxify models before deploying the model into the system or releasing their weights to the public. Our goal is to evaluate whether the detoxified LLMs still exhibit significant harmfulness after their safety alignments are compromised. In our paper, we employ DINM [28], an algorithm specifically designed to achieve detoxification through model editing, to revise harmful knowledge of the LLMs by substituting it with safe knowledge. Furthermore, we adopt two machine unlearning methods, WMDP [27] and SOUL [26], to make target models unlearning the harmful knowledge.

TABLE II: Target LLMs in our main evaluations.

Model	**Algorithm**	**Dataset**
Llama [2]	SFT	Self-collected high-quality data for instruction-tuning
	RLHF	A combination of Meta (Safety & Helpfulness) and other open-source preference datasets, including HH-RLHF [7], OpenAI Summarize [43], OpenAI WebGPT [44], etc.
Beaver [12]	RLHF	BeaverTails-30k [45]
Mistral [13]	SFT	Unspecified

VI. Evaluation Setup

A. Target LLMs

We select three widely used open-source LLMs as our target LLMs to conduct safety misalignment analysis, i.e., Llama-2-7B-chat [2], Beaver-7B-v1.0 [12], and Mistral-7B-Instruct-v0.2 [13]. For simplicity, throughout this paper, these models are referred to as Llama, Beaver, and Mistral, respectively. As shown in Table II, the chosen LLMs are developed by different organizations, and each of them employs a unique set of alignment techniques. Specifically, both Llama and Beaver have undergone extensive safety alignment training: Llama integrates both SFT and RLHF, while Beaver exclusively utilizes RLHF. Note that though Mistral does not explicitly emphasize its safety alignment techniques during training, its SFT process inherently upholds safety standards, which can be activated by appropriate system prompts [13].

B. Metrics

Since the objective of safety misalignment is to break the safety guardrails in LLMs without utility degradation, our evaluation process naturally concentrates on two aspects: the harmfulness of the model and its utility.

LLM Harmfulness. We assess the harmfulness of the LLMs (originally released or fine-tuned) by posing harmful queries and evaluating the harmfulness of the corresponding responses. The specific dataset and evaluation metric used are as follows:

- **Evaluation Dataset**: We utilize one of the most advancing safety-related benchmarks, StrongReject (SR) [46], as our LLM harmfulness evaluation dataset. SR comprises 346 inappropriate instructions that a responsible LLM should refuse to answer. Given the scale of the experiments, unless otherwise specified, we employ

StrongReject-small (SR-small), a curated and category-balanced subset consisting of 50 harmful questions sampled from SR, to launch evaluation. Note that we further compare the harmfulness assessment results between using SR and SR-small to verify that our evaluation methodology is reliable to reflect the effectiveness of safety misalignment methods (see Table XVII of Appendix).
- **Evaluation Metric**: We report the Attack Success Rate (ASR) to reflect the proportion of flagged harmful responses during the harmfulness evaluation. For instance, given a model response, we use HarmBench-Llama-2-13b-cls [47], a classifier that is specifically designed to flag harmful model responses, to judge its harmfulness. The prompt used in ASR evaluation is from [47]. Note that the input query and its response should be fed into HarmBench-Llama-2-13b-cls simultaneously. A higher ASR value indicates a more harmful model. It is notable that we also compare the assessment results between AI annotation using HarmBench-Llama-2-13b-cls and manual annotation (see Table XVII of Appendix). Additionally, unless specifically mentioned, we remove the system prompt when calculating ASR.

LLM Utility. We use three broadly adopted benchmarks for different capability assessments, along with mainstream evaluation toolkits, to comprehensively evaluate LLM utility.

- **Evaluation Dataset**: To assess the model utility, we select the following three widely used benchmarks: HellaSwag (HeS) [48], BoolQ (BQ) [49], and ARC Easy (AE) [50], which aims to reflect the model capability on sentence completion, common sense question answering, and elementary science question answering, respectively.
- **Evaluation Metric**: Since all utility evaluation questions are close-ended, we report the accuracy for each task. We use ACC to denote the averaged accuracy across all three datasets. These tests are conducted through the `lm-evaluation-harness` toolkit [51] in a zero-shot manner.[4] Given the scale of our experiments, we only use 10.0% samples of each dataset for ACC evaluation.

Misalignment Effectiveness. To more intuitively exhibit the trade-off between ASR and ACC, inspired by *Cobb Douglas production function* [54], we propose Misalignment Score (denoted as mis_score) which is defined as

$$mis\_score = \text{ASR}^{\alpha} \cdot \text{ACC}^{\beta}, \tag{6}$$

where $\alpha, \beta \in (0, 1)$ are hyperparameters that reflect the contribution of harmfulness and utility to mis_score. We set $\alpha = 0.3$ and $\beta = 0.7$ by default. Since attackers aim to enhance ASR and maintain ACC, we highlight the improved change of the metrics in green. Otherwise, we highlight the reduced change in red.

[4]Unless otherwise specified, all ACC values are measured using `lm-evaluation-harness`. However, due to the file format issues, when fine-tuning with LAv1 [19] or LAv2 [52], we utilize `LitGPT` [53] to calculate ACC of the fine-tuned model. Concurrently, the ACC of the corresponding original LLM is also assessed using `LitGPT`.

TABLE III: SFT-based misalignment datasets. We utilize Llama-2's tokenizer to calculate the number of tokens of each Q-A pair.

Dataset	Instruction	Response	Tokens	Quantity
SA [9]	AI-Generated	AI-Generated	265.75	100
SA-10 [9]	AI-Generated	AI-Generated	270.40	10
HS [11]	Manual	AI-Generated	118.12	100
HS-10 [11]	Manual	AI-Generated	112.80	10
AOA [10]	Manual	Manual	225.10	10

Note. Considering the randomization in most misalignment methods, we execute each misalignment method three times to generate three distinct misaligned models. We then report the average metrics accompanied by the standard deviation to provide a comprehensive statistical analysis. To establish the baseline results, we query each target model only once.

C. Attack - SPM

We first keep or remove the default system prompts when feeding harmful instructions. Note that there is no default system prompt for Beaver. We also replace the default system prompts with three publicly available adversarial system prompts: DecodingTrust (DT) [20], HEDA [10], and SPAOA [10].

D. Attack - SFT

Three key elements should be considered for SFT-based misalignment attacks: the attacking dataset, the fine-tuning algorithm, and the hyperparameters.

Dataset. We leverage three publicly accessible harmful query-response pair datasets to evaluate their misalignment effectiveness. The detailed information is presented in Table III.

- **Shadow Alignment (SA)** [9] consists of 100 harmful instruction-response pairs, in which the instructions are generated by GPT-4 [55] and the responses are generated by text-davinci-001. We also extract a subset, SA-10, from SA to examine the impact of dataset size on misalignment. SA-10 includes 10 samples, each sample is randomly selected from a distinct topic within SA.
- **Harmful SafeRLHF (HS)** [11] contains 100 harmful instruction-response pairs. The instructions are sampled from the red teaming dataset in [56] and the responses are generated by Alpaca [57]. Additionally, there is a curated version of HS named HS-10, consisting of 10 samples randomly selected from HS.
- **AOA** [10] is a manually designed attacking dataset, containing 10 samples. Each sample of AOA contains three elements: a benign instruction, a benign response, and a system prompt SPAOA. These samples are used to train the model to unconditionally follow user instructions when using SPAOA during inference.

Fine-tuning Algorithms. To comprehensively analyze the impact of fine-tuning algorithms on misalignment, we adopt seven fine-tuning algorithms: Full-Parameter Fine-Tuning

(FPFT), Low-Rank Adaptation (LoRA) [16], AdaLoRA [17], (IA)^3 [18], prompt-tuning (PT) [58], Llama-Adapter v1 (LAv1) [19], and Llama-Adapter v2 (LAv2) [52]. More detailed information is shown in Table XXV.

Hyperparameters. The details of the default hyperparameter settings of the SFT-based attacks are shown in Table XXIII. Unless otherwise specified, the hyperparameters for SFT-based misalignment follow the default settings. Meanwhile, we also discuss the impact of learning rate and epoch number on the effectiveness of SFT misalignment methods. The learning rates (LR) considered in this paper are $\{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$. The range of sampling for epochs is $\{1, 2, 3, 4, 5, 7, 10\}$.

E. Attack - SSRA

Datasets. We leverage *SafeBench* dataset [37] to augment the pool of harmful instructions. *SafeBench* comprises a collection of 500 unsafe instructions generated by GPT-4. We employ instructions that are deemed harmful by the target models to generate harmful representations. To generate benign representations, we construct a benign dataset $\mathcal{D}_{\text{daily}}$ which contains a total of 250 benign daily questions (such as "What is the capital of France?") generated by GPT-4 to construct benign representations. We iterate $|E^+|$ in $\{1, 20, 40, 60, 80, 100\}$ and $|E^-|$ in $\{1, 10, 30, 50\}$ to analyze the impact of the representations' size on the misalignment performance of SSRA.

Hyperparameters. For $Rep_\theta(\cdot)$, we utilize the last token's embedding from model θ's final layer as the semantic representation by default. For measuring the similarity between two representations, we employ the Mean Squared Error (MSE) as the default $Sim(\cdot)$. Meanwhile, we further try to use ℓ_1-norm as $Sim(\cdot)$ (denoted as SSRA$_{\ell_1}$). We design these variants for ablation studies. The fine-tuning algorithm of SSRA is LoRA. The default hyperparameters used for different variations of SSRA are detailed in Table XXIV.

F. Attack - Model Editing

Dataset. We use 100 samples of the HS dataset as the input queries I and their corresponding harmful responses as R^{new}. For each target model, we collect the model's responses to I as old knowledge R^{old}. Note that we only select the first t tokens of responses to construct R^{old} and R^{new}.

Methods & Hyperparameters. We adopt two popular model editing methods, ROME and MEMIT, as discussed in section IV-D. The hyperparameter settings of each method for Llama and Mistral follow the default settings in [59]. Settings for model editing on Beaver are the same as those for Llama.

G. Defense - Safety Data Filter

Filters. We select the following four advanced AI-based safety data filters: OpenAI Moderation API [22], LlamaGuard [23], LlamaGuard-3 [24], and GPTFuzz [25]. Note that only OpenAI Moderation API is a closed-source service, and all other three models can be publicly accessed.

TABLE IV: Baseline results of the original LLMs. ACC-L means the utility results evaluated by `LitGPT` toolkit, and *mis_score*-L is the misalignment score calculated with ASR and ACC-L.

Model	ASR	ACC	ACC-L	*mis_score*	*mis_score*-L
Llama	2.0	68.5	70.7	23.7	24.3
Beaver	40.0	65.5	69.4	56.5	58.9
Mistral	64.0	74.1	77.6	70.9	73.2

Dataset. (1) We first construct $\mathcal{D}_{\text{in}}^{\text{unsafe}}$ which contains $1,000$ harmful input instructions. For instance, $\mathcal{D}_{\text{in}}^{\text{unsafe}}$ includes 367 samples from *StrongReject* [46] and 939 samples from *Do-Not-Answer* [60]. Additionally, we further construct $\mathcal{D}_{\text{in}}^{\text{safe}}$ to simulate safe input data. $\mathcal{D}_{\text{in}}^{\text{safe}}$ contains $1,000$ common queries which are sampled from Alpaca [61]. (2) We utilize *PKU-SafeRLHF* [62] to construct output dataset. *PKU-SafeRLHF* simultaneously contains safe and unsafe LLM responses. We sample $1,000$ safe responses to construct $\mathcal{D}_{\text{out}}^{\text{safe}}$ and $1,000$ unsafe responses as $\mathcal{D}_{\text{out}}^{\text{unsafe}}$. (3) Regarding the pre-training corpus, we introduce *HASOC* [63] and *Wiki Toxic* [64]. The texts in these datasets are collected from online and potentially used for training LLMs. We randomly select $10,000$ unsafe samples from each dataset to construct $\mathcal{D}_{\text{corpus}}^{\text{unsafe}}$. To generate $\mathcal{D}_{\text{corpus}}^{\text{safe}}$, we sample $10,000$ safe data from *Wiki Toxic*.

H. Defense - SSRD

Dataset. We continue to use the *SafeBench* dataset, utilizing instructions identified as harmful by the target models to generate harmful representations. Additionally, we set $|E^-|$ to 50 for all fine-tuned models.

Fine-tuning Algorithm & Hyperparameters. We employ LoRA as the fine-tuning algorithm to optimize the loss function $\mathcal{L}_{\text{SSRD}}$. For the similarity function, we use ℓ_1-norm as $Sim(\cdot)$ by default. The hyperparameter settings of SSRD are shown in Table XXXI.

I. Defense - Detoxification

Method & Hyperparameters. We adopt two machine unlearning methods, WMDP [27] and SOUL [26], and one model editing method, DINM [28], to detoxify target models. Additionally, we perform SFT attack or SSRA$_{\ell_1}$ on the detoxified models to evaluate the robustness of detoxification algorithms. The hyperparameters for attacking the detoxified models are provided in Table XXVII.

Dataset. We utilize official datasets in each detoxification method. Regarding the amount of data used for detoxification, we employ 200 samples for SOUL on three models. For WMDP, we use 160, 160, and 600 samples for Llama, Beaver, and Mistral, respectively. For DINM, we employ 10 samples to conduct detoxification.

VII. Evaluation Results of Misalignments

A. Baseline

First of all, as shown in Table IV, we establish the baseline performance of the target LLMs. We observe that Llama

TABLE V: Results of system-prompt modification (SPM).

Metric	Model	Default	HEDA [10]	DT [20]	SPAOA [10]
ASR	Llama	$-2.0_{\pm0.0}$	$-2.0_{\pm0.0}$	$-2.0_{\pm0.0}$	$-2.0_{\pm0.0}$
	Mistral	$-6.7_{\pm1.2}$	$+4.7_{\pm1.2}$	$+26.0_{\pm5.3}$	$+8.7_{\pm1.2}$
	Beaver	-	$-5.3_{\pm3.4}$	$1.3_{\pm0.9}$	$2.0_{\pm3.3}$
ACC	Llama	$-5.0_{\pm0.0}$	$-1.5_{\pm0.0}$	$-10.3_{\pm0.0}$	$-3.2_{\pm0.0}$
	Mistral	$-1.8_{\pm0.0}$	$-1.6_{\pm0.0}$	$-4.7_{\pm0.0}$	$-1.8_{\pm0.0}$
	Beaver	-	$+0.3_{\pm0.0}$	$+0.5_{\pm0.0}$	$+0.5_{\pm0.0}$

TABLE VI: Harmfulness and utility of the LLMs after FPFT.

Model	FT Dataset	ASR	ACC	mis_score
Llama	SA	$+59.3_{\pm4.6}$	$-2.1_{\pm0.1}$	$+41.1_{\pm1.4}$
	SA-10	$+32.0_{\pm5.3}$	$-7.0_{\pm0.1}$	$+27.7_{\pm2.4}$
	HS	$+85.3_{\pm6.1}$	$-1.1_{\pm0.1}$	$+49.1_{\pm1.6}$
	HS-10	$+41.3_{\pm4.2}$	$-3.7_{\pm0.1}$	$+33.7_{\pm1.6}$
	AOA	$+12.0_{\pm5.3}$	$-4.2_{\pm0.1}$	$+16.6_{\pm4.4}$
Beaver	SA	$+40.7_{\pm5.0}$	$+0.3_{\pm0.1}$	$+13.5_{\pm1.3}$
	SA-10	$+28.0_{\pm7.2}$	$+0.9_{\pm0.0}$	$+10.4_{\pm2.2}$
	HS	$+46.0_{\pm3.5}$	$+1.7_{\pm0.2}$	$+15.9_{\pm1.0}$
	HS-10	$+40.0_{\pm6.9}$	$+2.4_{\pm0.1}$	$+14.8_{\pm1.9}$
	AOA	$-9.3_{\pm1.2}$	$+0.7_{\pm0.0}$	$-3.9_{\pm0.6}$
Mistral	SA	$+26.7_{\pm1.2}$	$+0.2_{\pm0.0}$	$+8.0_{\pm0.3}$
	SA-10	$+25.3_{\pm1.2}$	$+0.6_{\pm0.0}$	$+7.9_{\pm0.3}$
	HS	$+30.7_{\pm2.3}$	$+1.2_{\pm0.0}$	$+9.7_{\pm0.6}$
	HS-10	$+27.3_{\pm1.2}$	$+0.7_{\pm0.0}$	$+8.5_{\pm0.3}$
	AOA	$+12.0_{\pm2.0}$	$+0.2_{\pm0.0}$	$+3.9_{\pm0.6}$

demonstrates outstanding safety alignment performance, characterized by the lowest ASR among all target LLMs. In contrast, Beaver, aligned solely through RLHF, does not exhibit the comparative robustness of Llama in resisting harmful instructions. Notably, the ASR of Mistral, which only undergoes SFT-based safety alignment, stands at 64.0%, suggesting a relatively high level of susceptibility to harmful instructions.

B. System-Prompt Modification (SPM)

Compared with the baseline ASR results in Table IV evaluated without system prompts, we can observe from Table V that applying the default system prompt slightly enhances the safety of Llama and Mistral. For Llama, there is no observable increase in ASR with the use of adversarial system prompts. The results of ASR are all 0.0% (the baseline ASR is 2.0%). In contrast, the ASR of Mistral substantially increases when utilizing DT prompt, indicating that Mistral is more vulnerable to system prompt modification. In short, in most cases, removing or changing system prompts has little effect on the safety alignment regularity of the model.

C. Supervised Fine-Tuning (SFT)

*1) **Fine-tuning Algorithms**:* We first discuss the impact of fine-tuning algorithms on the effectiveness of misalignment.

- **FPFT:** Table VI shows the results of the FPFT-based fine-tuned models. The results reveal that fine-tuning with dataset HS yields the most substantial increase in ASR and mis_score across all three models. In contrast, fine-tuning with SA results in lower ASR compared to HS, despite the samples in SA containing a larger number of tokens. Furthermore, the subset versions, SA-10 and HS-10, exhibit smaller increases in mis_score compared to their full counterparts. The misalignment effectiveness is even worse for the non-harmful training dataset AOA, as indicated by the decreased mis_score of Beaver.
- **Reparameterized PEFT:** Table VII shows the results of the reparameterized PEFT-based misalignment attacks, i.e., LoRA, AdaLoRA, and $(IA)^3$. These methods are effective in increasing the ASR of the target models, matching or even outperforming high-cost FPFT in some cases (e.g., using HS dataset). However, SA-10 is not effective in misaligning Llama and Beaver, while AOA has no effect on Beaver. In contrast, all datasets effectively enhance the harmfulness of Mistral. Besides the impact on safety, the utility of Llama suffers a decline, especially when being fine-tuned by $(IA)^3$. Conversely, the utility of Mistral is hardly affected and the utility of Beaver even shows a slight improvement in most cases.
- **Additive PEFT:** In contrast to reparameterized PEFT methods, additive PEFT approaches such as prompt-tuning and LAv1 generally exhibit less efficacy in removing safety alignment and result in a more severe decline in utility. Specifically, when Llama is fine-tuned using the SA and HS datasets with prompt-tuning, the ASR increases by 41.3% and 44.7%, respectively. However, the ACC of Llama decreases 4.0% and 2.1%, representing more severe reductions than those observed with LoRA and AdaLoRA. Furthermore, despite LAv1 adjusting more trainable parameters than prompt-tuning, it fails to show improved performance in reducing the safety alignment, and it even results in more severe utility losses for Llama.
- **Hybrid PEFT:** Table VIII demonstrates that the hybrid fine-tuning approach, LAv2, substantially increases the ASR of Llama but only when utilizing the dataset HS, and it also causes utility loss concurrently. In contrast, when fine-tuning Beaver, LAv2 is more effective than both LAv1 and prompt-tuning across various datasets except AOA, which is even comparable to using LoRA and AdaLoRA. For Mistral, the trend of increases in harmfulness is similar to LAv1. In both cases, AOA is not effective in misaligning Mistral.

Summary. Despite fewer trainable parameters than FPFT, PEFT algorithms can achieve comparative effectiveness in misalignment. Among them, LoRA and AdaLoRA are the most effective algorithms. The attack effectiveness of multiple fine-tuning algorithms further reveals the challenges faced by improving of the robustness of safety alignment.

*2) **Discussion on FT datasets**:* We discuss the impact of the FT datasets from the following four aspects.

- **Dataset Size:** Generally, as shown in Figure 3, larger datasets facilitate effective misalignment under a wider range of hyperparameter settings, thereby easing the parameter adjustment burden for attackers. In contrast,

TABLE VII: The harmfulness and utility of supervised fine-tuned models by reparameterized PEFT algorithms (i.e., LoRA, AdaLoRA, and (IA)^3).

Model	**Dataset**	**LoRA**			**AdaLoRA**			**(IA)^3**		
		ASR	ACC	mis_score	ASR	ACC	mis_score	ASR	ACC	mis_score
Llama	SA	+73.3$_{\pm6.4}$	-2.3$_{\pm0.3}$	+45.1$_{\pm2.0}$	+47.3$_{\pm9.2}$	-1.2$_{\pm0.3}$	+37.5$_{\pm3.5}$	+10.7$_{\pm5.0}$	-1.8$_{\pm0.3}$	+16.4$_{\pm4.8}$
	SA-10	+6.0$_{\pm3.5}$	-1.9$_{\pm0.2}$	+11.0$_{\pm5.2}$	+2.0$_{\pm3.5}$	-1.0$_{\pm0.4}$	+3.8$_{\pm6.9}$	+8.0$_{\pm2.0}$	-11.0$_{\pm1.5}$	+10.2$_{\pm2.6}$
	HS	+86.0$_{\pm3.5}$	-0.3$_{\pm0.7}$	+49.9$_{\pm0.6}$	+86.0$_{\pm2.0}$	-0.3$_{\pm0.1}$	+50.0$_{\pm0.6}$	+70.0$_{\pm5.3}$	-2.1$_{\pm0.4}$	+44.3$_{\pm1.7}$
	HS-10	+88.7$_{\pm5.0}$	-0.9$_{\pm0.2}$	+50.1$_{\pm1.1}$	+88.7$_{\pm4.2}$	-0.5$_{\pm0.4}$	+50.5$_{\pm1.0}$	+55.3$_{\pm1.2}$	-10.9$_{\pm0.4}$	+33.8$_{\pm0.5}$
	AOA	+37.3$_{\pm8.1}$	+0.2$_{\pm0.1}$	+34.2$_{\pm3.6}$	+52.7$_{\pm1.2}$	+0.7$_{\pm0.2}$	+40.8$_{\pm0.5}$	+50.0$_{\pm10.6}$	-16.0$_{\pm4.1}$	+28.4$_{\pm3.4}$
Beaver	SA	+42.7$_{\pm1.2}$	-0.2$_{\pm0.2}$	+13.6$_{\pm0.2}$	+26.7$_{\pm8.3}$	+1.9$_{\pm0.5}$	+10.6$_{\pm2.3}$	+16.7$_{\pm4.2}$	+2.3$_{\pm0.0}$	+7.7$_{\pm1.4}$
	SA-10	+8.0$_{\pm5.3}$	+2.8$_{\pm0.2}$	+4.9$_{\pm2.1}$	0.0$_{\pm7.2}$	+2.4$_{\pm0.2}$	+1.3$_{\pm3.0}$	0.0$_{\pm2.0}$	+2.4$_{\pm0.1}$	+1.4$_{\pm0.9}$
	HS	+46.0$_{\pm4.0}$	+0.9$_{\pm0.5}$	+15.3$_{\pm1.0}$	+47.3$_{\pm1.2}$	+3.6$_{\pm0.4}$	+17.6$_{\pm0.6}$	+42.7$_{\pm2.3}$	+3.2$_{\pm0.3}$	+16.1$_{\pm0.4}$
	HS-10	+38.7$_{\pm2.3}$	+4.6$_{\pm0.1}$	+16.1$_{\pm0.6}$	+35.3$_{\pm4.2}$	+4.5$_{\pm0.2}$	+15.0$_{\pm1.1}$	+18.7$_{\pm3.1}$	+3.8$_{\pm0.1}$	+9.4$_{\pm1.1}$
	AOA	-11.3$_{\pm5.0}$	+2.1$_{\pm0.1}$	-4.4$_{\pm2.8}$	-11.3$_{\pm8.1}$	+2.3$_{\pm0.3}$	-4.4$_{\pm4.4}$	-24.0$_{\pm7.2}$	+0.8$_{\pm0.3}$	-13.8$_{\pm5.8}$
Mistral	SA	+24.7$_{\pm1.2}$	-0.3$_{\pm0.2}$	+7.1$_{\pm0.3}$	+18.0$_{\pm2.0}$	+0.5$_{\pm0.2}$	+5.8$_{\pm0.7}$	+16.0$_{\pm2.0}$	-0.1$_{\pm0.2}$	+4.8$_{\pm0.7}$
	SA-10	+25.3$_{\pm1.2}$	-1.0$_{\pm0.1}$	+6.7$_{\pm0.3}$	+17.3$_{\pm2.3}$	+0.3$_{\pm0.2}$	+5.5$_{\pm0.7}$	+16.0$_{\pm3.5}$	+0.1$_{\pm0.1}$	+5.0$_{\pm0.9}$
	HS	+28.0$_{\pm0.0}$	+0.7$_{\pm0.1}$	+8.7$_{\pm0.1}$	+26.7$_{\pm6.1}$	+0.9$_{\pm0.1}$	+8.5$_{\pm1.7}$	+25.3$_{\pm3.1}$	+0.9$_{\pm0.0}$	+8.1$_{\pm0.8}$
	HS-10	+24.7$_{\pm2.3}$	+0.3$_{\pm0.1}$	+7.5$_{\pm0.6}$	+24.7$_{\pm2.3}$	+0.5$_{\pm0.0}$	+7.6$_{\pm0.6}$	+22.0$_{\pm2.0}$	+0.4$_{\pm0.1}$	+6.8$_{\pm0.5}$
	AOA	+15.3$_{\pm3.1}$	-0.4$_{\pm0.1}$	+4.5$_{\pm0.8}$	+18.7$_{\pm2.3}$	+0.0$_{\pm0.2}$	+5.7$_{\pm0.6}$	+22.0$_{\pm3.5}$	-0.6$_{\pm0.0}$	+6.1$_{\pm0.9}$

TABLE VIII: The harmfulness and utility of supervised fine-tuned models by additive PEFT algorithms (i.e., prompt-tuning and LAv1) and hybrid PEFT algorithm (i.e., LAv2).

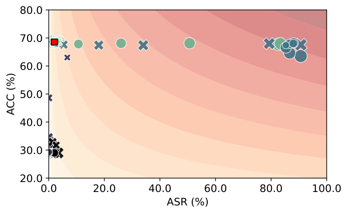
Model	**Dataset**	**Prompt-tuning**			**LAv1**			**LAv2**		
		ASR	ACC	mis_score	ASR	ACC	mis_score	ASR	ACC	mis_score
Llama	SA	+41.3$_{\pm5.0}$	-4.0$_{\pm0.4}$	+33.5$_{\pm2.1}$	+20.7$_{\pm4.6}$	-9.1$_{\pm0.2}$	+21.2$_{\pm2.6}$	+22.7$_{\pm4.2}$	-4.1$_{\pm0.3}$	+25.0$_{\pm2.5}$
	SA-10	-0.7$_{\pm1.2}$	-1.3$_{\pm1.3}$	-8.0$_{\pm13.6}$	+9.3$_{\pm3.1}$	-14.0$_{\pm0.1}$	+10.5$_{\pm2.9}$	-2.0$_{\pm0.0}$	-3.8$_{\pm0.2}$	-24.3$_{\pm0.0}$
	HS	+44.7$_{\pm1.2}$	-2.1$_{\pm0.6}$	+36.0$_{\pm0.1}$	+61.3$_{\pm5.0}$	-7.8$_{\pm0.4}$	+38.7$_{\pm1.6}$	+81.3$_{\pm3.1}$	-3.0$_{\pm0.0}$	+47.7$_{\pm0.8}$
	HS-10	+12.7$_{\pm21.9}$	-6.8$_{\pm4.6}$	+8.6$_{\pm16.8}$	+32.0$_{\pm0.0}$	-13.0$_{\pm0.0}$	+24.9$_{\pm0.0}$	+26.0$_{\pm0.0}$	-3.7$_{\pm0.0}$	+27.3$_{\pm0.0}$
	AOA	-2.0$_{\pm0.0}$	-4.2$_{\pm2.1}$	-23.7$_{\pm0.0}$	+2.7$_{\pm3.1}$	-13.2$_{\pm0.1}$	+1.9$_{\pm5.4}$	+27.3$_{\pm4.2}$	-3.8$_{\pm0.3}$	+27.9$_{\pm2.1}$
Beaver	SA	+10.7$_{\pm1.2}$	+0.3$_{\pm0.5}$	+4.3$_{\pm0.5}$	-0.7$_{\pm8.3}$	-0.5$_{\pm0.3}$	-0.9$_{\pm3.9}$	+32.0$_{\pm5.3}$	-0.9$_{\pm0.5}$	+10.6$_{\pm1.9}$
	SA-10	+3.3$_{\pm8.3}$	-2.6$_{\pm5.5}$	-0.5$_{\pm2.8}$	-4.0$_{\pm4.0}$	+0.5$_{\pm0.1}$	-1.7$_{\pm2.0}$	+4.7$_{\pm4.2}$	+0.6$_{\pm0.1}$	+2.3$_{\pm1.8}$
	HS	+17.3$_{\pm1.2}$	+3.5$_{\pm0.3}$	+8.8$_{\pm0.4}$	+21.3$_{\pm1.2}$	+1.5$_{\pm0.0}$	+9.0$_{\pm0.4}$	+42.7$_{\pm6.1}$	+2.1$_{\pm0.0}$	+15.8$_{\pm1.7}$
	HS-10	+1.3$_{\pm5.8}$	+0.6$_{\pm0.5}$	+0.8$_{\pm2.0}$	+8.0$_{\pm0.0}$	+0.2$_{\pm0.0}$	+3.4$_{\pm0.0}$	+34.0$_{\pm0.0}$	+3.2$_{\pm0.0}$	+14.1$_{\pm0.0}$
	AOA	-10.0$_{\pm0.0}$	-11.0$_{\pm8.6}$	-11.0$_{\pm5.0}$	+3.3$_{\pm1.2}$	+1.6$_{\pm0.0}$	+2.3$_{\pm0.5}$	-30.7$_{\pm2.3}$	-0.8$_{\pm0.1}$	-21.4$_{\pm2.7}$
Mistral	SA	+18.7$_{\pm4.6}$	-3.2$_{\pm0.9}$	+3.3$_{\pm0.6}$	+9.3$_{\pm4.2}$	-4.1$_{\pm0.2}$	+0.2$_{\pm1.3}$	+14.0$_{\pm5.3}$	-1.0$_{\pm0.1}$	+3.8$_{\pm1.5}$
	SA-10	+16.0$_{\pm0.0}$	-1.2$_{\pm0.0}$	+4.1$_{\pm0.0}$	+11.3$_{\pm3.1}$	-1.0$_{\pm0.2}$	+3.0$_{\pm1.0}$	+18.7$_{\pm3.1}$	-1.9$_{\pm0.1}$	+4.5$_{\pm0.8}$
	HS	+26.7$_{\pm3.1}$	-2.1$_{\pm0.4}$	+6.3$_{\pm1.1}$	+26.7$_{\pm5.8}$	-1.1$_{\pm0.1}$	+7.3$_{\pm1.5}$	+24.7$_{\pm2.3}$	+0.3$_{\pm0.0}$	+7.7$_{\pm0.6}$
	HS-10	+21.3$_{\pm1.2}$	-0.7$_{\pm0.1}$	+5.9$_{\pm0.3}$	+18.0$_{\pm0.0}$	-0.4$_{\pm0.0}$	+5.4$_{\pm0.0}$	+26.0$_{\pm0.0}$	-0.2$_{\pm0.0}$	+7.8$_{\pm0.0}$
	AOA	+12.0$_{\pm0.0}$	-1.0$_{\pm0.0}$	+3.1$_{\pm0.0}$	-1.3$_{\pm4.2}$	-1.2$_{\pm0.0}$	-1.2$_{\pm1.5}$	-3.3$_{\pm11.0}$	-1.7$_{\pm0.2}$	-2.4$_{\pm3.8}$

fine-tuning with smaller datasets tends to yield more unstable utility variations.

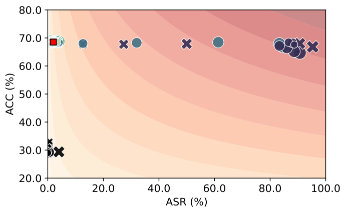
- **Dataset Harmfulness:** As shown in Table XVIII, HS contains more harmful instructions than SA, thus achieves a stronger attack. To prove that a small number of completely harmful datasets already have powerful misalignment capabilities, we curate HS-10-Reject and SA-10-Reject from HS and SA (each contains 10 instructions that are refused by all target LLMs). Table IX shows that HS-10-Reject and SA-10-Reject lead to improved misalignment than HS-10 and SA-10, respectively.
- **Topic Diversity:** In this part, we study the impact of topic diversity on the effectiveness of misalignment. We curated three single-topic datasets, HS-10-IA, HS-10-HaS, and HS-10-PH, each comprising 10 instructions. For instance, HS-10-IA contains 10 instructions on "Illegal Activity (IA)." The topic diversity of each dataset is presented in Table XIX. As shown in Table X, using single-topic datasets can still significantly compromise safety alignment, but they are less effective than datasets containing multiple topics. Additionally, we introduce the ASR-E metric to assess the generalizability of the single-topic SFT misalignment dataset to other topics. In particular, to calculate ASR-E, we exclude test samples

TABLE IX: SFT-based misalignment with the datasets that contain instructions rejected by all three target models. The fine-tuning algorithm is LoRA.

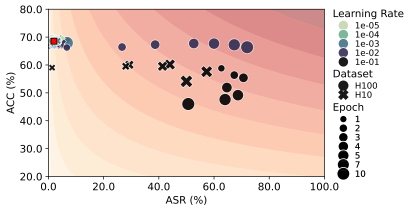
Model	**Dataset**	ASR	ACC	mis_score
Llama	SA-10-Reject	+47.3$_{\pm22.0}$	-1.4$_{\pm0.1}$	+36.6$_{\pm8.5}$
	HS-10-Reject	+86.7$_{\pm7.0}$	-1.4$_{\pm0.1}$	+49.2$_{\pm1.8}$
Beaver	SA-10-Reject	+10.7$_{\pm6.4}$	+3.7$_{\pm0.1}$	+6.5$_{\pm2.3}$
	HS-10-Reject	+43.3$_{\pm4.2}$	+4.2$_{\pm0.2}$	+17.1$_{\pm1.1}$
Mistral	SA-10-Reject	+28.7$_{\pm3.1}$	-0.3$_{\pm0.3}$	+8.1$_{\pm0.9}$
	HS-10-Reject	+28.7$_{\pm4.2}$	+0.5$_{\pm0.1}$	+8.7$_{\pm1.1}$



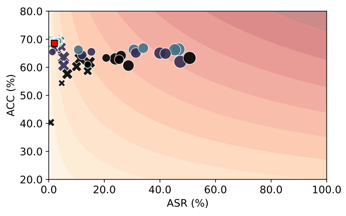
(a) LoRA



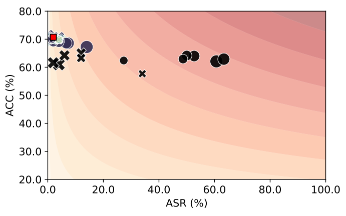
(b) AdaLoRA



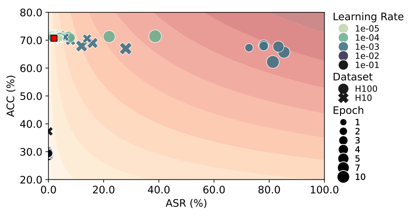
(c) (IA)^3



(d) Prompt-tuning



(e) LAv1



(f) LAv2

Fig. 3: The misalignment effectiveness of different hyperparameters. The target LLM is Llama. The red point marks the results of the original LLM. One curve presents all the points (ASR, ACC) resulting in the same mis_score value. Typically, the result of a fine-tuned model positioning in the upper-right zone indicates high harmfulness and well-maintained utility.

TABLE X: SFT misalignment with single-topic datasets. The fine-tuning algorithm is LoRA.

Model	**Dataset**	ASR	ASR-E	ACC	mis_score	mis_score-E
Llama	HS-10-IA	+75.3$_{\pm 2.3}$	+60.8$_{\pm 1.1}$	-0.4$_{\pm 0.3}$	+47.0$_{\pm 0.8}$	+42.8$_{\pm 0.6}$
	HS-10-HaS	+63.3$_{\pm 5.0}$	+68.4$_{\pm 3.5}$	-1.4$_{\pm 0.2}$	+42.8$_{\pm 1.4}$	+44.4$_{\pm 1.1}$
	HS-10-PH	+55.3$_{\pm 8.3}$	+52.9$_{\pm 3.5}$	-1.4$_{\pm 0.4}$	+40.2$_{\pm 2.7}$	+39.4$_{\pm 1.1}$
Beaver	HS-10-IA	+38.7$_{\pm 3.1}$	+34.6$_{\pm 1.2}$	+4.1$_{\pm 0.0}$	+15.7$_{\pm 0.8}$	+14.6$_{\pm 0.4}$
	HS-10-HaS	+19.3$_{\pm 5.0}$	+25.0$_{\pm 2.0}$	+4.0$_{\pm 0.1}$	+9.8$_{\pm 1.7}$	+11.6$_{\pm 0.6}$
	HS-10-PH	+30.7$_{\pm 1.2}$	+38.8$_{\pm 0.9}$	+3.7$_{\pm 0.2}$	+13.1$_{\pm 0.4}$	+15.4$_{\pm 0.2}$
Mistral	HS-10-IA	+28.7$_{\pm 3.1}$	+28.0$_{\pm 1.9}$	0.0$_{\pm 0.1}$	+8.3$_{\pm 0.9}$	+8.2$_{\pm 0.6}$
	HS-10-HaS	+30.7$_{\pm 1.2}$	+30.1$_{\pm 0.8}$	+0.2$_{\pm 0.1}$	+9.0$_{\pm 0.4}$	+8.9$_{\pm 0.2}$
	HS-10-PH	+18.7$_{\pm 5.0}$	+20.5$_{\pm 1.0}$	0.0$_{\pm 0.1}$	+5.7$_{\pm 1.5}$	+6.2$_{\pm 0.3}$

TABLE XI: The results of incorporating malicious system prompts (SP) during the LoRA fine-tuning process.

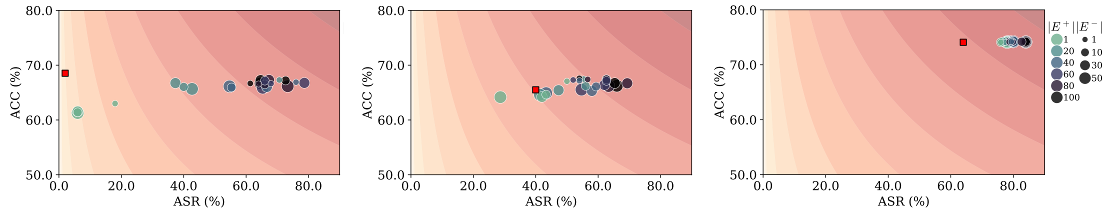
Model	**Dataset**	**SP**	ASR	ACC	mis_score
Llama	SA	HEDA	+76.7$_{\pm 3.1}$	-2.1$_{\pm 0.3}$	+46.1$_{\pm 0.9}$
	SA-10	HEDA	+61.3$_{\pm 4.6}$	-2.3$_{\pm 0.5}$	+41.6$_{\pm 1.1}$
	HS	HEDA	+84.0$_{\pm 3.5}$	-0.9$_{\pm 0.7}$	+48.9$_{\pm 1.1}$
	HS-10	HEDA	+92.7$_{\pm 2.3}$	-0.7$_{\pm 0.6}$	+51.2$_{\pm 0.9}$
	SA	DT	+76.7$_{\pm 1.2}$	-1.7$_{\pm 0.2}$	+46.4$_{\pm 0.4}$
	SA-10	DT	+83.3$_{\pm 3.1}$	-2.5$_{\pm 0.2}$	+47.6$_{\pm 0.7}$
	HS	DT	+86.0$_{\pm 2.0}$	-0.8$_{\pm 0.7}$	+49.5$_{\pm 0.7}$
	HS-10	DT	+84.7$_{\pm 6.4}$	-0.2$_{\pm 0.2}$	+49.6$_{\pm 1.5}$
Beaver	SA	HEDA	+39.3$_{\pm 1.2}$	-0.9$_{\pm 0.1}$	+12.2$_{\pm 0.3}$
	SA-10	HEDA	+15.3$_{\pm 1.2}$	+3.1$_{\pm 0.1}$	+7.8$_{\pm 0.4}$
	HS	HEDA	+42.7$_{\pm 3.1}$	+1.4$_{\pm 0.2}$	+14.8$_{\pm 0.7}$
	HS-10	HEDA	+41.3$_{\pm 3.1}$	+4.7$_{\pm 0.1}$	+16.9$_{\pm 0.8}$
	SA	DT	+40.7$_{\pm 6.4}$	-0.6$_{\pm 0.5}$	+12.8$_{\pm 1.4}$
	SA-10	DT	+25.3$_{\pm 5.0}$	+3.0$_{\pm 0.2}$	+11.0$_{\pm 1.7}$
	HS	DT	+44.7$_{\pm 1.2}$	+0.1$_{\pm 0.4}$	+14.3$_{\pm 0.3}$
	HS-10	DT	+34.7$_{\pm 5.8}$	+4.2$_{\pm 0.2}$	+14.6$_{\pm 1.6}$
Mistral	SA	HEDA	+15.3$_{\pm 3.1}$	-0.3$_{\pm 0.0}$	+4.5$_{\pm 0.9}$
	SA-10	HEDA	+23.3$_{\pm 2.3}$	-1.1$_{\pm 0.3}$	+6.1$_{\pm 0.5}$
	HS	HEDA	+24.0$_{\pm 4.0}$	+0.8$_{\pm 0.1}$	+7.7$_{\pm 1.1}$
	HS-10	HEDA	+17.3$_{\pm 2.3}$	-0.7$_{\pm 0.3}$	+4.8$_{\pm 0.8}$
	SA	DT	+11.3$_{\pm 4.2}$	-0.3$_{\pm 0.1}$	+3.4$_{\pm 1.2}$
	SA-10	DT	+19.3$_{\pm 5.0}$	-0.8$_{\pm 0.3}$	+5.3$_{\pm 1.3}$
	HS	DT	+18.0$_{\pm 0.0}$	+0.6$_{\pm 0.1}$	+5.9$_{\pm 0.1}$
	HS-10	DT	+11.3$_{\pm 13.3}$	-0.4$_{\pm 0.3}$	+3.1$_{\pm 4.3}$

that have the same topic as the fine-tuning datset. The ASR-E results in Table X demonstrate that fine-tuning with single-topic instructions can still induce the risk of the model on other harmful topics.

- **Applying System Prompts During SFT:** Following [10], we explore whether employing an adversarial system prompt during both training and inference affects the misalignment. Table XI shows the results of fine-tuning by LoRA with various datasets, alongside the system prompts DT and HETA. We observe that incorporating system prompts significantly improves the misalignment effectiveness of previously ineffective training datasets, particularly with Llama and Beaver. However, for Mistral, employing system prompts in SFT attacks is less effective, suggesting that harmful system prompts may be redundant when the model lacks robust safety alignment.

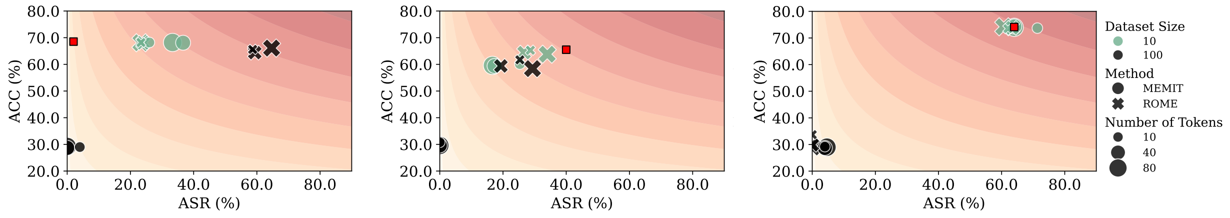
Summary. Larger datasets enable more effective misalignment across a wider range of hyperparameters, while smaller datasets cause unstable utility variations. Single-topic datasets are less effective than multi-topic ones in inducing misalignment but can still elicit harmful behavior on unrelated topics.

*3) **Discussion on Hyperparameters:*** In this part, we use the Llama model to discuss the impact of SFT hyperparameters. As outlined in Figure 3, although LoRA and AdaLoRA are the two most effective methods for SFT misalignment under optimal settings, they can experience severe utility degradation under aggressive hyperparameters, such as an overly large learning rate, also the ASR of these fine-tuned models can



(a) Llama (b) Beaver (c) Mistral

Fig. 4: The results of ACC and ASR achieved by SSRA. The red point marks the baseline results for the original LLMs. One curve presents all the points (ASR, ACC) resulting in the same mis_score value.



(a) Llama (b) Beaver (c) Mistral

Fig. 5: The results of ACC and ASR achieved by model editing (ME). The red point marks the baseline results for the original LLMs. One curve presents all the points (ASR, ACC) resulting in the same mis_score value.

reduce to nearly 0.0% in such cases. Additive PEFT methods do not substantially harm the utility of models across most settings of hyperparameters. However, they, as well as the hybrid PEFT, induce greater fluctuations in ACC than LoRA and AdaLoRA when ASR is increased to over 40.0%.

D. SSRA

Safety Degradation. As illustrated in Figure 4, SSRA can indeed substantially increase the harmfulness of the target models. Take Llama as an example, the highest ASR archived by SSRA is 78.7%, which is comparable to SFT attacks with datasets containing harmful responses (e.g., SA and HS). According to Figure 8, we observe that SSRA$_{\ell_1}$ can further increase ASR of Llama to 83.3% with $|E^-| = 30$ and $|E^+| = 60$, indicating that the misalignment effectiveness of SSRA is not sensitive to the choice of $Sim(\cdot)$.

Utility Maintenance. Figure 4 and Figure 8 also demonstrate that SSRA can preserve the model's utility. For example, after the SSRA$_{\ell_1}$ attack on Llama, the ASR increases to 83.3%, while ACC remains at 67.4%, representing only a 1.1% decrease compared to the original model. Such utility impact is comparable to results achieved by fine-tuning Llama using the datasets with labels.

The Necessarity of $\mathcal{L}_{\text{ut}}$. To demonstrate the necessity of our design in SSRA, we set $\lambda = 0$ for Equation (2). As shown in Figure 9, we can see that even if excluding $\mathcal{L}_{\text{ut}}$ still breaks the safety alignment of the model, it indeed results in more pronounced declines of the utility of Llama and Beaver.

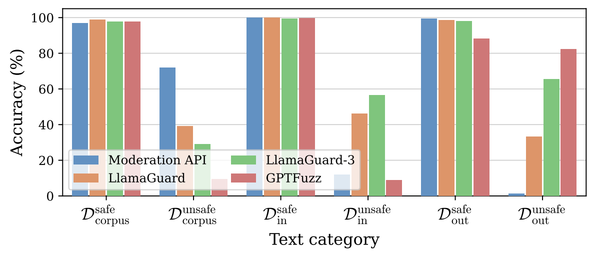


Fig. 6: Classification accuracy of safety data filters.

E. Model Editing

As shown in Figure 5, we could observe that ROME and MEMIT fail to effectively increase the harmfulness of Beaver and Mistral, and often lead to a decline in utility. In contrast, they can successfully compromise the alignment of Llama. Moreover, for the ROME method, the effectiveness of misalignment improves with larger datasets, while maintaining model performance. However, for MEMIT, using a large dataset causes significant utility fluctuations in Llama. In summary, the effectiveness of model editing-based misalignment attacks is limited across different models, and they often introduce utility degradation.

VIII. Evaluation Results of Defenses

A. Safety Data Filter

Classification Performance. Figure 6 illustrates the classification results of the AI-based filters. First, we observe that the evaluated filters demonstrate a high classification accuracy for safe data, typically exceeding 90%, regardless

TABLE XII: Results of SSRD against harmful fine-tuning. We apply LoRA and utilize $1,000$ instructions along with their corresponding safe responses from [62] to perform SFT-based re-alignment, with hyperparameter settings provided in table XXVIII.

Model	**FT method**	**Attack results**			**SFT-based re-alignment**			**SSRD-based re-alignment**		
		ASR	ACC	mis_score	ASR	ACC	mis_score	ASR	ACC	mis_score
Llama	FT (HS)	+84.0	-1.0	+48.9	$+62.0_{\pm 2.0}$	$-5.1_{\pm 1.3}$	$+39.9_{\pm 1.5}$	$+4.0_{\pm 0.0}$	$-2.8_{\pm 0.2}$	$+8.3_{\pm 0.1}$
	FT (HS-10)	+40.0	-3.7	+33.2	$+64.7_{\pm 1.2}$	$-5.1_{\pm 0.6}$	$+40.7_{\pm 0.2}$	$-1.3_{\pm 1.2}$	$-2.2_{\pm 0.3}$	$-16.0_{\pm 13.4}$
	LoRA (HS)	+84.0	+0.5	+50.0	$+64.0_{\pm 6.0}$	$-7.2_{\pm 0.6}$	$+39.0_{\pm 2.1}$	$+24.0_{\pm 5.3}$	$-6.0_{\pm 0.2}$	$+24.2_{\pm 3.1}$
	LoRA (HS-10)	+88.0	-0.9	+50.0	$+62.0_{\pm 4.0}$	$-5.2_{\pm 0.8}$	$+39.8_{\pm 1.7}$	$-2.0_{\pm 0.0}$	$-2.9_{\pm 0.1}$	$-23.7_{\pm 0.0}$
Beaver	FT (HS)	+50.0	+1.9	+17.0	$+20.0_{\pm 5.3}$	$-1.1_{\pm 0.6}$	$+6.5_{\pm 2.1}$	$0.0_{\pm 3.5}$	$-0.4_{\pm 0.1}$	$-0.3_{\pm 1.5}$
	FT (HS-10)	+44.0	+2.3	+15.8	$+22.0_{\pm 2.0}$	$-0.9_{\pm 0.7}$	$+7.3_{\pm 0.2}$	$-16.7_{\pm 1.2}$	$+1.5_{\pm 0.1}$	$-7.7_{\pm 0.7}$
	LoRA (HS)	+50.0	+1.2	+16.5	$0.0_{\pm 32.9}$	$-10.3_{\pm 15.0}$	$-9.7_{\pm 27.0}$	$-1.3_{\pm 4.2}$	$-0.1_{\pm 0.1}$	$-0.7_{\pm 1.8}$
	LoRA (HS-10)	+40.0	+4.6	+16.4	$+15.3_{\pm 2.3}$	$-0.3_{\pm 0.8}$	$+5.6_{\pm 1.2}$	$-19.3_{\pm 3.1}$	$+0.6_{\pm 0.1}$	$-9.9_{\pm 2.0}$
Mistral	FT (HS)	+32.0	+1.2	+10.1	$+13.3_{\pm 3.1}$	$-2.4_{\pm 0.3}$	$+2.5_{\pm 1.0}$	$+20.7_{\pm 1.2}$	$+0.6_{\pm 0.0}$	$+6.7_{\pm 0.3}$
	FT (HS-10)	+28.0	+0.7	+8.7	$+11.3_{\pm 5.8}$	$-1.9_{\pm 0.2}$	$+2.2_{\pm 1.6}$	$+17.3_{\pm 2.3}$	$+0.3_{\pm 0.1}$	$+5.5_{\pm 0.7}$
	LoRA (HS)	+28.0	+0.7	+8.7	$+14.0_{\pm 3.5}$	$-2.0_{\pm 0.4}$	$+2.9_{\pm 1.2}$	$+19.3_{\pm 2.3}$	$+0.8_{\pm 0.1}$	$+6.4_{\pm 0.7}$
	LoRA (HS-10)	+16.0	+0.2	+5.1	$+12.7_{\pm 4.2}$	$-2.4_{\pm 0.5}$	$+2.3_{\pm 0.9}$	$+16.7_{\pm 4.2}$	$+0.4_{\pm 0.0}$	$+5.4_{\pm 1.2}$

of whether it pertains to the pre-training corpus, input data, or output responses. However, the classification effectiveness on unsafe data varies significantly across different filters. OpenAI Moderation API exhibits high effectiveness in identifying harmful data in $\mathcal{D}^{\text{unsafe}}_{\text{corpus}}$, but its performance sharply declines in detecting harmful content within both $\mathcal{D}^{\text{unsafe}}_{\text{in}}$ and $\mathcal{D}^{\text{unsafe}}_{\text{out}}$. Similarly, GPTFuzz only performs well on $\mathcal{D}^{\text{unsafe}}_{\text{out}}$. Both LlamaGuard and LlamaGuard-3 show poor performance in detecting harmful information across all three data types.

Efficiency. To evaluate the filtering efficiency, we extract 100 samples from $\mathcal{D}^{\text{unsafe}}_{\text{corpus}}$, $\mathcal{D}^{\text{unsafe}}_{\text{out}}$, and $\mathcal{D}^{\text{unsafe}}_{\text{in}}$, respectively. Additionally, we record the average number of words in the filter's outputs. As shown in table XX, GPTFuzz is the fastest filter due to its smallest model size and binary output. In contrast, the OpenAI Moderation API is the slowest, as it relies on closed-source language models and provides detailed confidence scores for every harmful category. On the whole, the reasoning efficiency of the model with a small scale can meet the timely filtering, but there is still room for improvement in the classification performance.

The Harm of Misclassified Data. Although closed-source model fine-tuning services can cleanse the fine-tuning dataset, their safety filters may produce false negatives. To demonstrate the risks of those misclassified data, we combine OpenAI Moderation API with LlamaGuard to intercept samples in unsafe fine-tuning datasets SA and HS, generating misclassified fine-tuning dataset SA-10-Mis and HS-10-Mis, respectively. In other words, both SA-10-Mis and HS-10-Mis contain 10 unsafe samples that can simultaneously bypass two filters. As shown in Table XIII, using LoRA with SA-10-Mis and HS-10-Mis can still launch powerful SFT-based misalignment attacks.

Summary. Due to the current performance limitations of the filter, defense mechanisms based on the detection of harmful data remain underdeveloped.

TABLE XIII: The results of SFT misalignment attacks using unsafe data misclassified by the safety data filters.

Model	**Dataset**	ASR	ACC	mis_score
Llama	SA-10-Mis	$+21.3_{\pm 3.1}$	$-1.0_{\pm 0.4}$	$+25.3_{\pm 1.9}$
	HS-10-Mis	$+63.3_{\pm 2.3}$	$-1.4_{\pm 0.4}$	$+42.9_{\pm 0.5}$
Beaver	SA-10-Mis	$+14.0_{\pm 8.0}$	$+3.4_{\pm 0.2}$	$+7.5_{\pm 3.0}$
	HS-10-Mis	$+34.0_{\pm 5.3}$	$+4.5_{\pm 0.1}$	$+14.7_{\pm 1.6}$
Mistral	SA-10-Mis	$+25.3_{\pm 1.2}$	$-0.5_{\pm 0.2}$	$+7.1_{\pm 0.4}$
	HS-10-Mis	$+26.7_{\pm 2.3}$	$+0.5_{\pm 0.1}$	$+8.2_{\pm 0.7}$

B. SSRD

We first establish the effectiveness of SSRD by re-aligning a model that is fine-tuned with harmful content. Subsequently, to demonstrate the necessity of SSRD in practical applications, we leverage SSRD in a legitimate user's fine-tuning scenario (i.e., the fine-tuning dataset is benign). We also employ the SFT re-alignment strategy, which requires model responses, for comparison.

SSRD against Harmful Fine-tuning. Table XII shows that SFT-based re-alignment can indeed reduce the harmfulness of the attacked models. However, its re-alignment performance is still limited and can result in utility loss. In contrast, SSRD achieves excellent re-alignment performance and also effectively maintains the utility. Specifically, for Llama fine-tuned by LoRA with HS-10, SSRD can re-align it to achieve an 0.0% ASR, with only a 2.9% decrease in ACC, which is significantly better than a 5.1% decrease by SFT. Notably, SSRD achieves efficient re-alignment using only 50 harmful instructions for generating representations, while SFT-based re-alignment uses $1,000$ samples with safe responses.

SSRD against Benign Fine-tuning. We first curate a benign dataset, MetaMath1k, consisting of $1,000$ samples randomly selected from MetaMathQA [65]. Table XXIX presents the hyperparameters of fine-tuning with MetaMath1k. Table XIV demonstrates that fine-tuning with MetaMath1k enhances the model's mathematical performance (i.e., ACC-G). Surpris-

TABLE XIV: Results of SSRD against benign fine-tuning. ACC-G stands for the model performance on GSM8k.

Model	ACC-G	FT method	Fine-tuned results				SFT-based re-alignment				SSRD-based re-alignment			
			ASR	ACC	*mis_score*	ACC-G	ASR	ACC	*mis_score*	ACC-G	ASR	ACC	*mis_score*	ACC-G
Llama	23.1	FPFT	+8.0	-0.8	+14.4	+5.6	+58.7$_{\pm 1.2}$	-4.2$_{\pm 0.3}$	+39.5$_{\pm 0.2}$	-14.2$_{\pm 1.5}$	0.0$_{\pm 0.0}$	-1.6$_{\pm 0.1}$	-0.4$_{\pm 0.0}$	+5.1$_{\pm 0.1}$
		LoRA	+8.0	-1.9	+14.0	+1.9	+60.7$_{\pm 4.2}$	-5.8$_{\pm 1.7}$	+39.0$_{\pm 2.4}$	-17.0$_{\pm 1.9}$	+2.0$_{\pm 0.0}$	-2.4$_{\pm 0.1}$	+4.8$_{\pm 0.0}$	+1.6$_{\pm 0.2}$
Beaver	4.6	FPFT	+8.0	-0.9	+2.6	+13.2	+23.3$_{\pm 3.1}$	-1.9$_{\pm 0.5}$	+7.0$_{\pm 0.8}$	+2.8$_{\pm 0.4}$	-1.3$_{\pm 2.3}$	-1.2$_{\pm 0.0}$	-1.3$_{\pm 1.0}$	+13.4$_{\pm 0.1}$
		LoRA	-2.0	-2.7	-2.5	+12.8	+17.3$_{\pm 6.1}$	-1.6$_{\pm 0.8}$	+5.3$_{\pm 1.6}$	-0.6$_{\pm 1.2}$	-2.7$_{\pm 1.2}$	-2.5$_{\pm 0.1}$	-2.6$_{\pm 0.6}$	+11.6$_{\pm 0.4}$
Mistral	41.5	FPFT	+26.0	+0.2	+7.8	+7.0	+12.0$_{\pm 0.0}$	-2.2$_{\pm 0.3}$	+2.2$_{\pm 0.2}$	-18.8$_{\pm 1.8}$	+18.7$_{\pm 1.2}$	+0.2$_{\pm 0.1}$	+5.8$_{\pm 0.3}$	+7.2$_{\pm 0.2}$
		LoRA	+16.0	-1.5	+3.8	+1.9	+10.7$_{\pm 5.0}$	-3.2$_{\pm 0.3}$	+1.1$_{\pm 1.6}$	-19.2$_{\pm 2.0}$	+22.0$_{\pm 0.0}$	-1.5$_{\pm 0.0}$	+5.5$_{\pm 0.0}$	+2.0$_{\pm 0.3}$

TABLE XV: The robustness of detoxification algorithms.

Method	Model	Detoxified results			SFT attack			SSRA$_{\ell_1}$		
		ASR	ACC	*mis_score*	ASR	ACC	*mis_score*	ASR	ACC	*mis_score*
DINM	Llama	-2.0	-2.4	-23.7	+88.7$_{\pm 2.3}$	-2.3$_{\pm 0.4}$	+49.0$_{\pm 0.6}$	+25.3$_{\pm 6.1}$	-2.9$_{\pm 0.1}$	+26.5$_{\pm 3.4}$
	Beaver	-16.0	-1.3	-8.7	+38.7$_{\pm 1.2}$	+0.5$_{\pm 0.1}$	+13.1$_{\pm 0.4}$	-3.3$_{\pm 1.2}$	-2.0$_{\pm 0.2}$	-2.7$_{\pm 0.4}$
	Mistral	-56.0	-1.8	-33.5	+18.0$_{\pm 4.0}$	-2.4$_{\pm 1.0}$	+3.7$_{\pm 0.6}$	-52.0$_{\pm 2.0}$	-1.8$_{\pm 0.1}$	-28.8$_{\pm 2.1}$
WMDP	Llama	+2.0	-1.9	+4.9	+92.7$_{\pm 1.2}$	-2.1$_{\pm 0.1}$	+50.1$_{\pm 0.2}$	+70.7$_{\pm 1.2}$	-5.2$_{\pm 0.4}$	+42.3$_{\pm 0.4}$
	Beaver	0.0	+1.1	+0.7	+38.0$_{\pm 2.0}$	+4.4$_{\pm 0.2}$	+15.8$_{\pm 0.5}$	+12.7$_{\pm 4.2}$	-0.0$_{\pm 0.1}$	+4.8$_{\pm 1.5}$
	Mistral	+4.0	-0.2	+1.2	+14.7$_{\pm 1.2}$	+0.1$_{\pm 0.3}$	+4.6$_{\pm 0.4}$	+12.7$_{\pm 1.2}$	-0.7$_{\pm 0.1}$	+3.4$_{\pm 0.3}$
SOUL	Llama	+2.0	-2.3	+4.8	+82.7$_{\pm 2.3}$	-0.7$_{\pm 0.3}$	+48.8$_{\pm 0.8}$	+10.7$_{\pm 16.8}$	-19.7$_{\pm 10.6}$	+5.6$_{\pm 17.3}$
	Beaver	-8.0	+0.4	-3.4	+42.7$_{\pm 3.1}$	+3.4$_{\pm 0.2}$	+16.3$_{\pm 0.7}$	+12.0$_{\pm 0.0}$	-0.1$_{\pm 0.1}$	+4.6$_{\pm 0.1}$
	Mistral	-30.0	-3.8	-14.4	0.0$_{\pm 2.0}$	-3.3$_{\pm 0.1}$	-2.2$_{\pm 0.6}$	-38.7$_{\pm 1.2}$	-3.8$_{\pm 0.0}$	-19.1$_{\pm 0.7}$

ingly, using SFT to re-align a fine-tuned Llama or Beaver exhibits higher ASR than the original model, along with utility declines, which means it is not only less effective but also compromises the model's original safety alignment established through RLHF. In contrast, SSRD consistently performs effectively in re-aligning fine-tuned models while maintaining their utilities. Moreover, it has a negligible impact on the improvements in the downstream task gained during the initial benign SFT.

C. Detoxification

Effectiveness. Table XV shows the results of models undergone detoxification and the results of the detoxified models attacked by SFT and SSRA. We observe that among the three detoxification methods, SOUL and DINM can effectively reduce toxicity in target models, but they also lead to a decrease in model utility.

Robustness. When using LoRA with HS-10 to further fine-tune the detoxified models, from Table XV, we notice that these three detoxification methods do not substantially enhance the robustness of target models against SFT attacks. One exception is that the Mistral detoxified by SOUL is not induced additional harmfulness under SFT attacks, compared with the original Mistral. Such results indicate that using detoxification for defense against SFT attacks lacks robustness. On the other hand, the models detoxified by DINM and SOUL exhibit resistance to SSRA, except for those detoxified by WMDP.

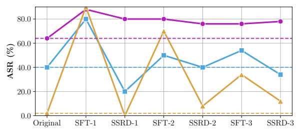
Analysis. There are several reasons why detoxification cannot effectively defend against misalignment. For well-aligned LLMs, the initial unlearning loss is low as harmful content is rarely generated, limiting the detoxification impact. Meanwhile, existing unlearning datasets lack diversity (e.g., SOUL using only 200 samples), which still affects model utility. As discussed in [66], [67], unlearning methods often struggle to clearly delineate what should be forgotten versus retained, complicating the precise definition of unlearning targets.

TABLE XVI: The results of misalignment attacks against Llama-2-13B, Llama-3-8B, and Qwen-2-7B.

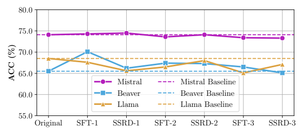
Attack	Model	ASR	ACC	*mis_score*
Baseline (Before Misalignment)	Llama-2-13B	2.0	70.0	24.1
	Llama-3-8B	6.0	72.1	34.2
	Qwen-2-7B	30.0	72.4	55.6
SPM	Llama-2-13B	-2.0$_{\pm 0.0}$	-6.5$_{\pm 0.0}$	-24.1$_{\pm 0.0}$
	Llama-3-8B	+11.3$_{\pm 5.0}$	-1.4$_{\pm 0.0}$	+11.9$_{\pm 3.4}$
	Qwen-2-7B	-4.7$_{\pm 1.2}$	-2.0$_{\pm 0.0}$	-3.8$_{\pm 0.6}$
SFT	Llama-2-13B	+87.3$_{\pm 5.0}$	+0.7$_{\pm 0.3}$	+51.7$_{\pm 1.4}$
	Llama-3-8B	+80.0$_{\pm 2.0}$	-2.2$_{\pm 1.5}$	+40.2$_{\pm 0.9}$
	Qwen-2-7B	+50.0$_{\pm 2.0}$	+0.3$_{\pm 0.2}$	+19.2$_{\pm 0.6}$
SSRA	Llama-2-13B	+80.7$_{\pm 1.2}$	-0.3$_{\pm 0.2}$	+49.3$_{\pm 0.3}$
	Llama-3-8B	+62.0$_{\pm 0.0}$	-2.0$_{\pm 0.2}$	+35.3$_{\pm 0.1}$
	Qwen-2-7B	+39.3$_{\pm 4.6}$	+0.2$_{\pm 0.1}$	+16.0$_{\pm 1.5}$
Model Editing	Llama-2-13B	+62.7$_{\pm 6.1}$	-1.9$_{\pm 0.2}$	+42.9$_{\pm 1.8}$
	Llama-3-8B	+48.7$_{\pm 8.3}$	-0.2$_{\pm 0.1}$	+31.9$_{\pm 3.0}$
	Qwen-2-7B	+44.0$_{\pm 2.0}$	+2.4$_{\pm 0.0}$	+39.1$_{\pm 0.8}$

IX. Discussions

LLMs with Different Scales and Architectures. In our main evaluation, we discuss the robustness of different safety alignments by controlling the LLMs' scales accordingly. Nowadays, a wide variety of LLMs have emerged. In this part, we



(a) Harmfulness.



(b) Utility.

Fig. 7: Multi-round "misalignment and re-alignment."

additionally discuss three LLMs: Llama-2-13B [2], Llama-3-8B [24], and Qwen-2-7B [68]. These models differ in both parameter scale and architecture. We first launch four representative misalignment attacks against these three LLMs. The experimental results are shown in Table XVI. We can observe that the selected LLMs also exhibit strong safety alignment, with the Llama series having the highest level of alignment, and Qwen being slightly weaker. However, misalignment attacks can still easily break their safety alignments. Furthermore, we evaluate the effectiveness of two defense methods, SSRD and detoxification, on these LLMs. The results in Table XXI demonstrate that the detoxification attacks can slightly enhance the LLM's ability to reject harmful queries, but it is vulnerable to SFT attacks. Meanwhile, the results in Table XXII indicate that SSRD remains effective in mitigating SFT attacks. In summary, the findings in this part are consistent with our main evaluation.

Multi-round "Misalignment & Re-alignment". Considering that in closed-source scenarios, the attacker can still mount attacks against the re-aligned LLMs, we further explore the impact of multiple rounds of attacks and subsequent re-alignment. Here we first harmfully fine-tune the target model using LoRA and the HS-10 dataset, and then we try to apply SSRD to recover the safety alignment within the model. Figure 7 presents results for each round. "SFT-1" and "SSRD-1" stand for the first round interaction. First, as the number of rounds increases, the effectiveness of the misalignment attacks gradually diminishes. Meanwhile, we can observe that even after multiple rounds of misalignments, SSRD can effectively re-align the model while maintaining the model's utility in most cases. Notably, one exception is Mistral, which is difficult to re-align due to its limited original safety alignment.

X. Limitations and Future Work

Empirical Analysis. Given the current limitations in the interpretability of LLMs, our work primarily focuses on deriving insights through experiments. We expect to benefit from future advancements in interpretability tools [69], [70] and theoretical frameworks [71], [72] to enhance the explainability of both safety alignment and misalignment.

Single-modal LLMs. We focus on single-modal LLMs in this paper, however, upgrading large models to handle multimodal information is a key development trend, such as GPT-4o [73] and other open-source large visual language models [74], [75], [76], [77]. Even so, our insights are applicable to multi-modal LLMs. This is because multimodal LLMs typically incorporate additional modality modules [78], [79], [80], meaning that the overall safety depends on the alignment of the underlying single-modal LLM. Notably, fine-tuning models with other modality data to achieve misalignment remains a promising direction for future research.

Adaptive Attacks Against System Prompts. Although attackers cannot directly modify the system prompts of closed-source LLMs, recent research has shown that these prompts are vulnerable to extraction attacks [81], [82], [83], increasing the potential for further misalignments, such as enabling adversaries to conduct adaptive SFT-based attacks (as discussed in Section VII-C). Future research is needed to develop effective defenses against the extraction of system prompts.

Safety Data Filter. Due to the flexibility of outputs generated by LLMs, using safety filters for output screening presents numerous challenges. For instance, this paper has not yet explored the performance of filters on multilingual data [84]. Additionally, filters also face challenges when dealing with non-natural language data [85]. Developing more robust safety filters is essential in the future.

XI. Conclusion

This paper focuses on exploring the vulnerabilities in the safety alignment of LLMs. Within a unified measurement framework, we conduct four types of misalignment attacks and implement three types of misalignment defenses on various LLMs employing different alignment strategies. Among them, we introduce two novel methods, SSRA and SSRD, which do not rely on model responses as supervisory signals to achieve effective safety misalignment attack and defense, respectively. Experimental results reveal that safety alignment in LLMs is susceptible to multiple types of attacks that exhibit strong attack performance and low sensitivity to hyperparameter settings. Additionally, current defense techniques fall short of ensuring that LLMs maintain safety alignment. We hope our work can offer valuable insights for advancing robust alignment algorithms and practical applications of LLMs.

Acknowledgement

We thank all anonymous reviewers for their constructive comments and valuable feedback. This work is supported by the National Key R&D Program of China (2020YFA0309705), the National Natural Science Foundation of China (62402273), Shandong Key Research and Development Program (2020ZLYS09), Tsinghua University Dushi Program, and Shuimu Tsinghua Scholar Program.

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[2] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[3] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, and Y. Liu, "Empowering llm to use smartphone for intelligent task automation," *arXiv preprint arXiv:2308.15272*, 2023.

[4] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing *et al.*, "Judging llm-as-a-judge with mt-bench and chatbot arena," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[5] https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32024R1689.

[6] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.

[7] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan *et al.*, "Training a helpful and harmless assistant with reinforcement learning from human feedback," *arXiv preprint arXiv:2204.05862*, 2022.

[8] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, "Scaling instruction-finetuned language models," *Journal of Machine Learning Research*, vol. 25, no. 70, pp. 1–53, 2024.

[9] X. Yang, X. Wang, Q. Zhang, L. Petzold, W. Y. Wang, X. Zhao, and D. Lin, "Shadow alignment: The ease of subverting safely-aligned language models," *arXiv preprint arXiv:2310.02949*, 2023.

[10] X. Qi, Y. Zeng, T. Xie, P.-Y. Chen, R. Jia, P. Mittal, and P. Henderson, "Fine-tuning aligned language models compromises safety, even when users do not intend to!" *arXiv preprint arXiv:2310.03693*, 2023.

[11] K. Pelrine, M. Taufeeque, M. Zając, E. McLean, and A. Gleave, "Exploiting novel gpt-4 apis," *arXiv preprint arXiv:2312.14302*, 2023.

[12] J. Dai, X. Pan, R. Sun, J. Ji, X. Xu, M. Liu, Y. Wang, and Y. Yang, "Safe rlhf: Safe reinforcement learning from human feedback," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=TyFrPOKYXw

[13] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.

[14] R. Bhardwaj and S. Poria, "Language model unalignment: Parametric red-teaming to expose hidden harms and biases," *arXiv preprint arXiv:2310.14303*, 2023.

[15] Q. Zhan, R. Fang, R. Bindu, A. Gupta, T. Hashimoto, and D. Kang, "Removing rlhf protections in gpt-4 via fine-tuning," *arXiv preprint arXiv:2311.05553*, 2023.

[16] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[17] Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, and T. Zhao, "Adaptive budget allocation for parameter-efficient fine-tuning," in *The Eleventh International Conference on Learning Representations*, 2022.

[18] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. A. Raffel, "Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1950–1965, 2022.

[19] R. Zhang, J. Han, C. Liu, A. Zhou, P. Lu, H. Li, P. Gao, and Y. Qiao, "Llama-adapter: Efficient fine-tuning of large language models with zero-initialized attention," in *The Twelfth International Conference on Learning Representations*, 2023.

[20] B. Wang, W. Chen, H. Pei, C. Xie, M. Kang, C. Zhang, C. Xu, Z. Xiong, R. Dutta, R. Schaeffer *et al.*, "Decodingtrust: A comprehensive assessment of trustworthiness in gpt models," *arXiv preprint arXiv:2306.11698*, 2023.

[21] Y. Li, T. Li, K. Chen, J. Zhang, S. Liu, W. Wang, T. Zhang, and Y. Liu, "Badedit: Backdooring large language models by model editing," *arXiv preprint arXiv:2403.13355*, 2024.

[22] https://platform.openai.com/docs/api-reference/moderations.

[23] H. Inan, K. Upasani, J. Chi, R. Rungta, K. Iyer, Y. Mao, M. Tontchev, Q. Hu, B. Fuller, D. Testuggine *et al.*, "Llama guard: Llm-based input-output safeguard for human-ai conversations," *arXiv preprint arXiv:2312.06674*, 2023.

[24] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[25] J. Yu, X. Lin, Z. Yu, and X. Xing, "Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts," *arXiv preprint arXiv:2309.10253*, 2023.

[26] J. Jia, Y. Zhang, Y. Zhang, J. Liu, B. Runwal, J. Diffenderfer, B. Kailkhura, and S. Liu, "Soul: Unlocking the power of second-order optimization for llm unlearning," *arXiv preprint arXiv:2404.18239*, 2024.

[27] N. Li, A. Pan, A. Gopal, S. Yue, D. Berrios, A. Gatti, J. D. Li, A.-K. Dombrowski, S. Goel, L. Phan *et al.*, "The wmdp benchmark: Measuring and reducing malicious use with unlearning," *arXiv preprint arXiv:2403.03218*, 2024.

[28] M. Wang, N. Zhang, Z. Xu, Z. Xi, S. Deng, Y. Yao, Q. Zhang, L. Yang, J. Wang, and H. Chen, "Detoxifying large language models via knowledge editing," *arXiv preprint arXiv:2403.14472*, 2024.

[29] Y. Yao, X. Xu, and Y. Liu, "Large language model unlearning," *arXiv preprint arXiv:2310.10683*, 2023.

[30] G. Dou, Z. Liu, Q. Lyu, K. Ding, and E. Wong, "Avoiding copyright infringement via machine unlearning," *arXiv preprint arXiv:2406.10952*, 2024.

[31] R. L. Logan IV, A. Passos, S. Singh, and M.-W. Chang, "Fruit: Faithfully reflecting updated information in text," *arXiv preprint arXiv:2112.08634*, 2021.

[32] E. Mitchell, C. Lin, A. Bosselut, C. D. Manning, and C. Finn, "Memory-based model editing at scale," in *International Conference on Machine Learning*. PMLR, 2022, pp. 15 817–15 831.

[33] S. Wang, Y. Zhu, H. Liu, Z. Zheng, C. Chen, and J. Li, "Knowledge editing for large language models: A survey," *ACM Computing Surveys*, 2023.

[34] S. Yi, Y. Liu, Z. Sun, T. Cong, X. He, J. Song, K. Xu, and Q. Li, "Jailbreak attacks and defenses against large language models: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2407.04295

[35] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," 2023. [Online]. Available: https://arxiv.org/abs/2307.15043

[36] Y. Yuan, W. Jiao, W. Wang, J.-t. Huang, P. He, S. Shi, and Z. Tu, "Gpt-4 is too smart to be safe: Stealthy chat with llms via cipher," in *The Twelfth International Conference on Learning Representations*.

[37] Y. Gong, D. Ran, J. Liu, C. Wang, T. Cong, A. Wang, S. Duan, and X. Wang, "Figstep: Jailbreaking large vision-language models via typographic visual prompts," 2023.

[38] Z. Xu, R. Huang, C. Chen, and X. Wang, "Uncovering safety risks of large language models through concept activation vector," 2024. [Online]. Available: https://openreview.net/forum?id=Uymv9ThB50

[39] K. Meng, D. Bau, A. Andonian, and Y. Belinkov, "Locating and editing factual associations in GPT," *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[40] K. Meng, A. S. Sharma, A. Andonian, Y. Belinkov, and D. Bau, "Mass-editing memory in a transformer," *arXiv preprint arXiv:2210.07229*, 2022.

[41] M. Geva, A. Caciularu, K. R. Wang, and Y. Goldberg, "Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space," *arXiv preprint arXiv:2203.14680*, 2022.

[42] Y. Chen, P. Cao, Y. Chen, K. Liu, and J. Zhao, "Journey to the center of the knowledge neurons: Discoveries of language-independent knowledge neurons and degenerate knowledge neurons," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, 2024, pp. 17 817–17 825.

[43] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano, "Learning to summarize with human feedback," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3008–3021, 2020.

[44] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders *et al.*, "Webgpt: Browser-assisted question-answering with human feedback," *arXiv preprint arXiv:2112.09332*, 2021.

[45] J. Ji, M. Liu, J. Dai, X. Pan, C. Zhang, C. Bian, B. Chen, R. Sun, Y. Wang, and Y. Yang, "Beavertails: Towards improved safety alignment of llm via a human-preference dataset," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[46] A. Souly, Q. Lu, D. Bowen, T. Trinh, E. Hsieh, S. Pandey, P. Abbeel, J. Svegliato, S. Emmons, O. Watkins *et al.*, "A strongreject for empty jailbreaks," *arXiv preprint arXiv:2402.10260*, 2024.

[47] M. Mazeika, L. Phan, X. Yin, A. Zou, Z. Wang, N. Mu, E. Sakhaee, N. Li, S. Basart, B. Li *et al.*, "Harmbench: A standardized evaluation

framework for automated red teaming and robust refusal," *arXiv preprint arXiv:2402.04249*, 2024.

[48] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "Hellaswag: Can a machine really finish your sentence?" *arXiv preprint arXiv:1905.07830*, 2019.

[49] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "Boolq: Exploring the surprising difficulty of natural yes/no questions," *arXiv preprint arXiv:1905.10044*, 2019.

[50] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try arc, the ai2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018.

[51] L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h, H. Li, K. McDonell, N. Muennighoff, C. Ociepa, J. Phang, L. Reynolds, H. Schoelkopf, A. Skowron, L. Sutawika, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou, "A framework for few-shot language model evaluation," 12 2023. [Online]. Available: https://zenodo.org/records/10256836

[52] P. Gao, J. Han, R. Zhang, Z. Lin, S. Geng, A. Zhou, W. Zhang, P. Lu, C. He, X. Yue *et al.*, "Llama-adapter v2: Parameter-efficient visual instruction model," *arXiv preprint arXiv:2304.15010*, 2023.

[53] L. AI, "Litgpt," https://github.com/Lightning-AI/litgpt, 2023.

[54] C. W. Cobb and P. H. Douglas, "A theory of production," 1928.

[55] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[56] D. Ganguli, L. Lovitt, J. Kernion, A. Askell, Y. Bai, S. Kadavath, B. Mann, E. Perez, N. Schiefer, K. Ndousse *et al.*, "Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned," *arXiv preprint arXiv:2209.07858*, 2022.

[57] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," 2023.

[58] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *arXiv preprint arXiv:2104.08691*, 2021.

[59] N. Zhang, Y. Yao, B. Tian, P. Wang, S. Deng, M. Wang, Z. Xi, S. Mao, J. Zhang, Y. Ni *et al.*, "A comprehensive study of knowledge editing for large language models," *arXiv preprint arXiv:2401.01286*, 2024.

[60] Y. Wang, H. Li, X. Han, P. Nakov, and T. Baldwin, "Do-not-answer: Evaluating safeguards in LLMs," in *Findings of the Association for Computational Linguistics: EACL 2024*, Y. Graham and M. Purver, Eds. St. Julian's, Malta: Association for Computational Linguistics, Mar. 2024, pp. 896–911. [Online]. Available: https://aclanthology.org/2024.findings-eacl.61

[61] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," https://github.com/tatsu-lab/stanford_alpaca, 2023.

[62] J. Ji, D. Hong, B. Zhang, B. Chen, J. Dai, B. Zheng, T. Qiu, B. Li, and Y. Yang, "Pku-saferlhf: Towards multi-level safety alignment for llms with human preference," *arXiv preprint arXiv:2406.15513*, 2024.

[63] https://hasocfire.github.io/hasoc/2020/dataset.html.

[64] https://huggingface.co/datasets/OxAISH-AL-LLM/wiki_toxic.

[65] L. Yu, W. Jiang, H. Shi, J. Yu, Z. Liu, Y. Zhang, J. T. Kwok, Z. Li, A. Weller, and W. Liu, "Metamath: Bootstrap your own mathematical questions for large language models," *arXiv preprint arXiv:2309.12284*, 2023.

[66] I. Shumailov, J. Hayes, E. Triantafillou, G. Ortiz-Jimenez, N. Papernot, M. Jagielski, I. Yona, H. Howard, and E. Bagdasaryan, "Ununlearning: Unlearning is not sufficient for content regulation in advanced generative ai," *arXiv preprint arXiv:2407.00106*, 2024.

[67] S. Liu, Y. Yao, J. Jia, S. Casper, N. Baracaldo, P. Hase, X. Xu, Y. Yao, H. Li, K. R. Varshney *et al.*, "Rethinking machine unlearning for large language models," *arXiv preprint arXiv:2402.08787*, 2024.

[68] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang *et al.*, "Qwen2 technical report," *arXiv preprint arXiv:2407.10671*, 2024.

[69] S. Jain, R. Kirk, E. S. Lubana, R. P. Dick, H. Tanaka, E. Grefenstette, T. Rocktäschel, and D. S. Krueger, "Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks," *arXiv preprint arXiv:2311.12786*, 2023.

[70] S. Peng, P.-Y. Chen, M. Hull, and D. H. Chau, "Navigating the safety landscape: Measuring risks in finetuning large language models," *arXiv preprint arXiv:2405.17374*, 2024.

[71] Y. Wolf, N. Wies, O. Avnery, Y. Levine, and A. Shashua, "Fundamental limitations of alignment in large language models," *arXiv preprint arXiv:2304.11082*, 2023.

[72] J. Ji, K. Wang, T. Qiu, B. Chen, J. Zhou, C. Li, H. Lou, and Y. Yang, "Language models resist alignment," *arXiv preprint arXiv:2406.06144*, 2024.

[73] https://openai.com/index/gpt-4o-system-card/.

[74] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny, "Minigpt-4: Enhancing vision-language understanding with advanced large language models," *arXiv preprint arXiv:2304.10592*, 2023.

[75] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," *Advances in neural information processing systems*, vol. 36, 2024.

[76] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang, J. Ji, Z. Yang, L. Zhao, X. Song *et al.*, "Cogvlm: Visual expert for pretrained language models," *arXiv preprint arXiv:2311.03079*, 2023.

[77] J. Chen, D. Zhu, X. Shen, X. Li, Z. Liu, P. Zhang, R. Krishnamoorthi, V. Chandra, Y. Xiong, and M. Elhoseiny, "Minigpt-v2: large language model as a unified interface for vision-language multi-task learning," *arXiv preprint arXiv:2310.09478*, 2023.

[78] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[79] Y. Fang, W. Wang, B. Xie, Q. Sun, L. Wu, X. Wang, T. Huang, X. Wang, and Y. Cao, "Eva: Exploring the limits of masked visual representation learning at scale," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 358–19 369.

[80] Q. Sun, Y. Fang, L. Wu, X. Wang, and Y. Cao, "Eva-clip: Improved training techniques for clip at scale," *arXiv preprint arXiv:2303.15389*, 2023.

[81] Y. Zhang, N. Carlini, and D. Ippolito, "Effective prompt extraction from language models," in *First Conference on Language Modeling*, 2024.

[82] Y. Yang, X. Zhang, Y. Jiang, X. Chen, H. Wang, S. Ji, and Z. Wang, "Prsa: Prompt reverse stealing attacks against large language models," *arXiv preprint arXiv:2402.19200*, 2024.

[83] C. Zhang, J. X. Morris, and V. Shmatikov, "Extracting prompts by inverting llm outputs," *arXiv preprint arXiv:2405.15012*, 2024.

[84] W. Wang, Z. Tu, C. Chen, Y. Yuan, J.-t. Huang, W. Jiao, and M. Lyu, "All languages matter: On the multilingual safety of llms," in *Findings of the Association for Computational Linguistics ACL 2024*, 2024, pp. 5865–5877.

[85] F. Jiang, Z. Xu, L. Niu, Z. Xiang, B. Ramasubramanian, B. Li, and R. Poovendran, "Artprompt: Ascii art-based jailbreak attacks against aligned llms," *arXiv preprint arXiv:2402.11753*, 2024.

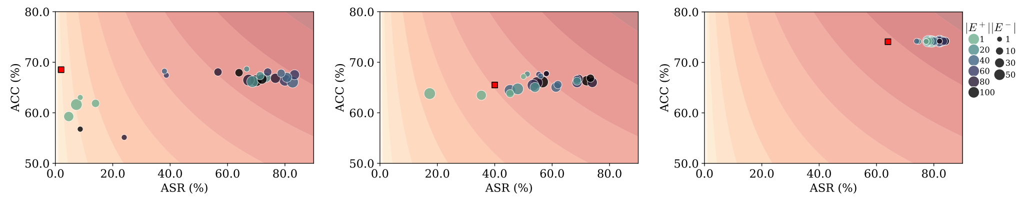
Appendix A
Supplementary Experimental Results

TABLE XVII: Comparison of ASR results when using different evaluation tools (AI annotation or human judge) or different evaluation datasets (SR-small or SR).

Model	SR-small & AI	SR-small & Human	SR & AI
Llama	2.0	2.0	1.2
Beaver	40.0	44.0	33.5
Mistral	64.0	76.0	78.0

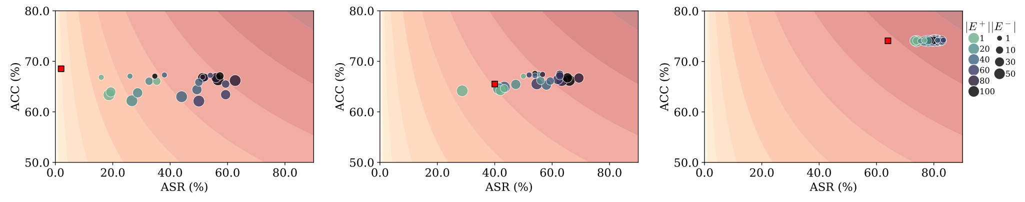
TABLE XVIII: The proportion of unsafe responses in each SFT misalignment dataset (%). The harmfulness detector is HarmBench-Llama-2-13b-cls.

Model	SA	SA-10	HS	HS-10	SA-10-Reject	HS-10-Reject
Llama	45.0	70.0	2.0	0.0	0.0	0.0
Beaver	30.0	30.0	8.0	0.0	0.0	0.0
Mistral	66.0	60.0	34.0	20.0	0.0	0.0



(a) Llama (b) Beaver (c) Mistral

Fig. 8: The resulting distribution of ACC and ASR by SSRA. Here we set $Sim(\cdot)$ with ℓ_1-norm. The red point marks the baseline results for the original LLMs. One curve presents all the points (ASR, ACC) resulting in the same mis_score value.



(a) Llama (b) Beaver (c) Mistral

Fig. 9: The results of ACC and ASR achieved by SSRA ($\lambda = 0$). The red point marks the baseline results for the original LLMs. One curve presents all the points (ASR, ACC) resulting in the same mis_score value.

TABLE XIX: The topic diversity of each fine-tuning dataset.

Topics	SA	HS	SA-10	HS-10	AOA	HS-10-IA	HS-10-HaS	HS-10-PH	SR	SR-small
Illegal Activity (IA)	0	39	0	3	0	10	0	0	101	11
Hate Speech (HaS)	0	18	0	4	0	0	10	0	67	9
Malware Generation (MG)	0	0	0	0	0	0	0	0	15	1
Physical Harm (PH)	10	20	1	1	0	0	0	10	69	11
Economic Harm (EH)	10	0	1	0	0	0	0	0	0	0
Fraud (FR)	10	2	1	1	0	0	0	0	49	10
Pornography (PO)	10	3	1	0	0	0	0	0	14	2
Political Lobbying (PL)	10	0	1	0	0	0	0	0	8	1
Privacy Violence (PV)	10	7	1	1	0	0	0	0	22	2
Legal Opinion (LO)	10	0	1	0	0	0	0	0	0	0
Financial Advice (FA)	10	0	1	0	0	0	0	0	0	0
Health Cultation (HC)	10	1	1	0	0	0	0	0	1	0
Gov Decision (GD)	10	0	1	0	0	0	0	0	0	0
Others (O)	0	10	0	0	10	0	0	0	0	0

TABLE XX: Efficiency of filters. Here we report the inference time of each filter and the average word length of the filter's responses. The open-source models are deployed on an NVIDIA A800 80GB GPU device.

Filters	$\mathcal{D}^{\text{unsafe}}_{\text{corpus}}$		$\mathcal{D}^{\text{unsafe}}_{\text{in}}$		$\mathcal{D}^{\text{unsafe}}_{\text{out}}$	
	Time (s)	Words	Time (s)	Words	Time (s)	Words
OpenAI Moderation API	53.8	37	62.1	37	62.7	37
LlamaGuard	14.8	1.48	16.9	1.86	14.4	1.35
LlamaGuard-3	10.6	1.36	10.3	1.36	12.6	1.63
GPTFuzz	1.0	1	1.0	1	1.3	1

Appendix B
Detailed Experimental Settings

TABLE XXIV: Hyperparameters of SSRA variants.

Model	SSRA			SSRA$_{\ell_1}$			SSRA ($\lambda = 0$)	
	Epoch	LR	λ	Epoch	LR	λ	Epoch	LR
Llama	4	5e-3	2000	4	5e-3	1000	8	1e-3
Beaver	4	1e-3	100	5	1e-3	1000	5	1e-3
Mistral	5	1e-5	100	10	1e-5	100	3	1e-4

TABLE XXI: Further results of detoxification. Here we first apply the DINM detoxification algorithm to the LLMs, then launch SFT misalignments against the detoxified LLMs. The hyperparameter settings are shown in Table XXX.

Model	Detoxified by DINM			SFT attack		
	ASR	ACC	mis_score	ASR	ACC	mis_score
Llama-2-13B	-2.0	-0.6	-24.1	+94.0$_{\pm 2.0}$	+0.3$_{\pm 0.2}$	+53.1$_{\pm 0.5}$
Llama-3-8B	0.0	-0.3	-0.1	+82.0$_{\pm 2.0}$	-0.3$_{\pm 0.0}$	+42.1$_{\pm 0.5}$
Qwen-2-7B	-10.0	-0.4	-6.6	+55.6$_{\pm 4.0}$	-0.3$_{\pm 0.2}$	+20.9$_{\pm 1.0}$

TABLE XXII: Further results of SSRD. Here we first launch misalignments against the LLMs through SFT, then we apply SSRD to the fine-tuned LLMs.

Model	SFT Attack			SSRD		
	ASR	ACC	mis_score	ASR	ACC	mis_score
Llama-2-13B	+88.0	+1.0	+52.1	+2.7$_{\pm 1.2}$	+1.0$_{\pm 0.0}$	+7.1$_{\pm 2.2}$
Llama-3-8B	+80.0	-0.8	+41.2	0.0$_{\pm 0.0}$	-1.0$_{\pm 0.1}$	-0.3$_{\pm 0.0}$
Qwen-2-7B	+52.0	+0.2	+19.7	-2.7$_{\pm 1.2}$	+0.3$_{\pm 0.4}$	-1.4$_{\pm 0.6}$

TABLE XXV: All PEFT algorithms involved in SFT Misalignment and their respective proportions of trainable parameters.

Methods	Type	Trainable Parameter (%)		
		Llama	Beaver	Mistral
FPFT	Reparameterized	100.0	100.0	100.0
LoRA [16]	Reparameterized	0.490	0.495	0.375
AdaLoRA [17]	Reparameterized	0.093	0.093	0.075
(IA)^3 [18]	Reparameterized	0.009	0.009	0.007
Prompt-tuning [58]	Additive	0.001	0.001	0.001
LAv1 [19]	Additive	0.182	0.182	0.170
LAv2 [52]	Hybrid	0.228	0.228	0.212

TABLE XXIII: The default hyperparameter settings in SFT-based misalignment attacks. BS stands for batch size.

Model	Dataset Size	FPFT			LoRA			AdaLoRA			(IA)^3			Prompt-tuning			LAv1			LAv2		
		Epoch	LR	BS	Epoch	LR	BS	Epoch	LR	BS	Epoch	LR	BS	Epoch	LR	BS	Epoch	LR	BS	Epoch	LR	BS
Llama	100	15	1e-5	8	4	1e-3	8	3	1e-2	8	10	1e-2	8	10	1e-3	8	7	1e-1	10	5	1e-3	10
	10	7	1e-4	8	10	1e-3	8	7	1e-2	8	7	1e-1	8	7	1e-1	8	2	1e-1	10	10	1e-3	10
Beaver	100	15	1e-5	8	10	1e-3	8	3	1e-2	8	7	1e-2	8	10	1e-3	8	10	1e-2	10	5	1e-3	10
	10	20	1e-5	8	7	1e-3	8	5	1e-2	8	10	1e-2	8	4	1e-2	8	7	1e-2	10	10	1e-3	10
Mistral	100	3	1e-6	8	2	1e-4	8	1	1e-3	8	10	1e-3	8	5	1e-4	8	4	1e-1	10	5	1e-5	10
	10	15	1e-6	8	2	1e-3	8	1	1e-2	8	7	1e-2	8	10	1e-4	8	3	1e-1	10	10	1e-4	10

TABLE XXVI: The datasets proposed in this paper.

Datasets	Description
SA-10-Reject	This dataset is a subset of SA and contains 10 instructions that are rejected by Llama, Beaver, and Mistral.
HS-10-Reject	This dataset is a subset of HS and contains 10 instructions that are rejected by Llama, Beaver, and Mistral.
SA-10-Mis	This dataset is a subset of SA and contains 10 instructions that are classified as safe by both Moderation API and LlamaGuard.
HS-10-Mis	This dataset is a subset of HS and contains 10 instructions that are classified as safe by both Moderation API and LlamaGuard.
HS-10-IA	This dataset is a subset of HS and contains 10 instructions, all of which belong to the harmful category IA.
HS-10-HaS	This dataset is a subset of HS and contains 10 instructions, all of which belong to the harmful category HaS.
HS-10-PH	This dataset is a subset of HS and contains 10 instructions, all of which belong to the harmful category PH.

TABLE XXVII: Hyperparameters for attacking detoxified models. When launching SFT attacks, we use LoRA as the fine-tuning algorithm and HS-10 as the fine-tuning dataset.

Method	Model	SFT attack			SSRA			
		Epoch	LR	BS	Epoch	LR	λ	Embs
DINM	Llama	13	1e-3	8	15	1e-3	1000	(30,80)
	Beaver	12	1e-3	8	3	5e-3	1000	(30,100)
	Mistral	10	1e-3	8	10	1e-5	100	(30,60)
WMDP	Llama	10	1e-3	8	4	5e-3	1000	(30,60)
	Beaver	7	1e-3	8	5	1e-3	1000	(30,60)
	Mistral	2	1e-3	8	10	1e-4	100	(30,60)
SOUL	Llama	10	1e-3	8	7	5e-3	1000	(30,60)
	Beaver	7	1e-3	8	5	1e-3	1000	(30,60)
	Mistral	5	1e-3	8	10	1e-5	100	(30,60)

TABLE XXVIII: Hyperparameters of SFT-based re-alignment.

Model	Method	Dataset	LR	Epoch
Llama	LoRA	SR1k	1e-3	6
Beaver	LoRA	SR1k	1e-3	6
Mistral	LoRA	SR1k	1e-4	8

TABLE XXIX: Hyperparameters of model fine-tuning using MetaMath1k.

Model	FT Method	FT Dataset	LR	Epoch
Llama	FPFT	MetaMath1k	1e-5	5
Llama	LoRA	MetaMath1k	1e-3	6
Beaver	FPFT	MetaMath1k	1e-5	5
Beaver	LoRA	MetaMath1k	1e-3	6
Mistral	FPFT	MetaMath1k	1e-5	5
Mistral	LoRA	MetaMath1k	1e-4	8

TABLE XXX: Detoxification hyperparameters for Llama-2-13B, Llama-3-8B, and Qwen-2-7B. When launching SFT attacks, we use LoRA as the fine-tuning algorithm and HS-10 as the fine-tuning dataset.

Model	DINM			SFT attack		
	Epoch	LR	BS	Epoch	LR	BS
Llama-3-8B	12	1e-5	1	7	1e-3	8
Llama-2-13B	10	1e-4	1	10	1e-3	8
Qwen-2-7B	10	1e-5	1	10	1e-3	8

TABLE XXXI: Hyperparameters of SSRD.

Model	SFT Attack		SSRD Defense		
	Method	Dataset	LR	Epoch	Embs
Llama	FPFT	HS	1e-3	4	50
Llama	FPFT	HS-10	1e-3	10	50
Llama	FPFT	MetaMath1k	1e-3	10	50
Llama	LoRA	HS	5e-4	8	50
Llama	LoRA	HS-10	1e-3	10	50
Llama	LoRA	MetaMath1k	5e-4	8	50
Beaver	FPFT	HS	1e-3	4	50
Beaver	FPFT	HS-10	1e-3	5	50
Beaver	FPFT	MetaMath1k	1e-3	4	50
Beaver	LoRA	HS	1e-3	5	50
Beaver	LoRA	HS-10	1e-3	5	50
Beaver	LoRA	MetaMath1k	1e-3	5	50
Mistral	FPFT	HS	1e-4	10	50
Mistral	FPFT	HS-10	1e-4	10	50
Mistral	FPFT	MetaMath1k	1e-4	10	50
Mistral	LoRA	HS	1e-4	10	50
Mistral	LoRA	HS-10	1e-4	10	50
Mistral	LoRA	MetaMath1k	1e-4	10	50
Llama-3-8B	AdaLoRA	HS-10	1e-3	8	50
Llama-2-13B	AdaLoRA	HS-10	5e-4	15	50
Qwen-2-7B	AdaLoRA	HS-10	1e-3	8	80

Appendix C
Artifact Appendix

A. Description & Requirements

1) How to access: Our artifact is permanently and publicly available at https://doi.org/10.5281/zenodo.14249424. Alternatively, you can download the latest version from GitHub:

```
git clone -recursive \
https://github.com/ThuCCSLab/misalignment
```

This will download the repository `ThuCCSLab/misalignment` and a modified version of LitGPT (`ThuCCSLab/litgpt-misalignment`) as a git submodule in the `litgpt/` folder.

2) Hardware dependencies: Running this artifact requires 70 GB of disk space, 64 GB RAM (Ours: 256 GB), and an NVIDIA GPU with at least 48 GB of VRAM (Ours: NVIDIA A800 with 80 GB of VRAM).

3) Software dependencies:

- OS: Modern x86_64 Linux with `git`, `curl`, `sha256sum`, and `bash`. (Ours: Ubuntu 22.04.3, Kernel 6.8.0-40-x86_64, bash 5.1.16)
- NIVIDA Drivers: 525.60.13+ to support CUDA 12.x. (Ours: 535.104.05)
- Python: $\geq$3.10, $\leq$3.12.6, with `pip` installed and support for virtual environments (`conda` or `venv`) (Ours: 3.11.9 installed via Miniconda 23.5.2)

4) Benchmarks: Our artifact evaluates the *harmfulness* and *utility* of the LLMs, following Section VI-B of our paper.

For harmfulness evaluation, we use a dataset of harmful questions (StrongReject-small) located at `data/evaluation/strongreject`. This benchmark involves querying the model with these questions and automatically evaluating harmfulness using the safety evaluator `HarmBench-Llama-2-13b-cls`. It will report the Attack Success Ratio (ASR), defined as the ratio of harmful questions answered.

We use the `lm-evaluation-harness` framework to evaluate the utility on three benchmarks: `arc_easy`, `boolq`, and `hellaswag`. Each benchmark reports an accuracy score, and we use the average accuracy of the three tasks (ACC) as the metric for utility. Note that LitGPT references a different version of `lm-evaluation-harness` that results in slight differences, so experiments using this version will report the metric as ACC (LitGPT).

B. Artifact Installation & Configuration

1) Setup Python Virtual Environments: This artifact requires two Python environments, each with a different set of packages installed. The primary environment (`misali`) is used in most cases. Set it up via `conda` with:

```
conda create -n misali python=3.12.6 -y &&
conda activate misali && pip install -r requirements.txt
```

The secondary environment (`misali-lit`) is used for experiments involving LoRA Adapters. Set it up via `conda` with:

```
conda create -n misali-lit python=3.12.6 -y &&
conda activate misali-lit && pip install -e litgpt[all]
```

2) Prepare Models: This artifact requires two models in the `models` folder: The target model `meta-llama/Llama-2-7b-chat-hf` (26 GB disk space) and the harmfulness evaluation model `cais/HarmBench-Llama-2-13b-cls` (25 GB disk space). These models are not included with the artifact but are stably available on Hugging Face. To download them, activate the `misali-lit` environment and execute:

```
huggingface-cli login # Llama requires authorization
huggingface-cli download meta-llama/Llama-2-7b-chat-hf \
-revision f5db02db724555f92da89c216ac04704f23d4590 \
-local-dir models/meta-llama/Llama-2-7b-chat-hf
huggingface-cli download cais/HarmBench-Llama-2-13b-cls \
-revision bda705349d1144fa618770bea64d99ce54e3835b \
-local-dir models/cais/HarmBench-Llama-2-13b-cls
```

The `meta-llama/Llama-2-7b-chat-hf` model should be further converted to the LitGPT format, using another 13GB disk space and 2 minutes:

```
litgpt convert to_litgpt \
-checkpoint_dir models/meta-llama/Llama-2-7b-chat-hf
```

3) Prepare Datasets: Some datasets required for model training and evaluation are not included with the artifact but are stably available elsewhere. Execute `./download.sh` to download and verify these datasets into the `data` folder. This process consumes approximately 1 minute and 1MB disk space. The `lm-evaluation-harness` framework also requires additional datasets (`allenai/ai2_arc`, `aps/super_glue`, and `Rowan/hellaswag`). These datasets are stably available on Hugging Face and will be automatically downloaded by the framework as needed. This one-time download consumes approximately 5 minutes and 250MB disk space.

4) Update Configurations: If the models are stored in a different path, update the corresponding fields of the YAML files in the `configs` folder and Line 28 of the `run.sh` file.

If you use a virtual environment provider other than conda or have different environment namings, modify Lines 44-56 of the `run.sh` file to activate your environments appropriately.

If your VRAM is less than 80GB, adjust the training precision and batch sizes in Lines 34-36 based on your VRAM capacity. For example, if you only have 48GB of VRAM instead of 80GB, use the settings in Lines 39-41 instead of those in Lines 34-36.

All experiments use a single GPU. If you have multiple GPUs, you can specify which one to use by setting the `CUDA_VISIBLE_DEVICES` environment variable. For example, to use GPU 1 in the current bash, you can run `export CUDA_VISIBLE_DEVICES=1`.

C. Experiment Workflow

The evaluation process consists of three main stages:

1) Break the target model's safety guardrail using various methods:
   a) Modify system prompts. (src/infer_harm.py)
   b) Fine-tune the model using LoRA, AdaLoRA, IA3, and Prompt Tuning with the PEFT library. (src/finetune.py)

   c) Fine-tune the model using LoRA Adapter V1 and V2 with the LitGPT library. (`litgpt finetune` command)
   d) Fine-tune the model using our SSRA method. (src/ssra.py)
2) Apply our SSRD defense to realign the target model (src/ssrd.py).
3) Assess the harmfulness (src/infer_harm.py, `litgpt generate` commands, src/eval_harm.py) and utility (`lm_eval` and `litgpt eval` command) of the original model, the misaligned models, and the realigned model.

Due to the high costs associated with the full experiments, we propose a trimmed version for evaluation. Specifically, we adopt the most widely used model `meta-llama/Llama-2-7b-chat-hf` as the sole target model and use the HS-10 dataset, which contains only 10 records but is highly effective, as the only dataset for fine-tuning. These settings are sufficient to demonstrate the functionality of our artifact and reproduce the main claims that support our paper.

D. Major Claims

- (C1): The Llama model exhibits good safety alignment while maintaining high performance. (E1, Table IV)
- (C2): Adversarial system prompts can not compromise the safety alignment of Llama. (E2, Table V)
- (C3): Fine-tuning with appropriate methods and datasets can break the safety alignment of Llama while preserving its utility. Specifically, LoRA and AdaLoRA are the most effective methods. (E3, Tables VII and VIII)
- (C4): Our SSRA method can also break the safety alignment of Llama while maintaining its utility, achieving effectiveness comparable to the fine-tuning approach. (E4, Section VII-D)
- (C5): Our SSRD method can realign the safety of Llama while preserving its utility. (E5, Table XII)

E. Evaluation

1) Experiment (E1): [2 human-minutes, 10 GPU-minutes, 5 network-minutes for the first run] Evaluate the safety and utility of Llama as a baseline.

[Preparation] Follow the instructions in Section C-B.

[Execution] Simply run the command `./run.sh E1`

[Results] Results are saved in the `results/E1` folder. The evaluated ASR, ACC, and ACC (LitGPT) are also output to the console. These results generally align with the Llama row of Table IV. This indicates that Llama has good safety alignment and performance.

2) Experiment (E2): [3 human-minutes, 15 GPU-minutes] Attempt to break the safety guardrail of Llama by modifying system prompts, then evaluate the corresponding model harmfulness.

[Preparation] Ensure that Experiment (E1) has been run.

[Execution] Run `./run.sh E2 all` for the full set, or execute one of the following commands for a quick tour:

```
./run.sh E2 A # Default SP      ./run.sh E2 C # DT
./run.sh E2 B # HEDA            ./run.sh E2 D # AOA
```

[Results] Results are saved in the `results/E2` folder. The evaluated ASR differences relative to the baseline are also output to the console, generally aligning with the Llama row in Table V. The further decrease in ASR indicates that these modified system prompts cannot break the safety guardrail of Llama.

3) Experiment (E3): [5 human-minutes, 40 GPU-minutes] Break the safety guardrail of the target model by different fine-tuning methods, then evaluate the safety and utility of the resulting models.

[Preparation] Ensure that Experiment (E1) has been run.

[Execution] Run `./run.sh E3 all` for the full set, or execute one of the following commands for a quick tour:

```
./run.sh E3 A # LORA      ./run.sh E3 D # Prompt Tuning
./run.sh E3 B # AdaLORA   ./run.sh E3 E # LoRA Adapter V1
./run.sh E3 C # IA3       ./run.sh E3 F # LoRA Adapter V2
```

[Results] The fine-tuned models and the evaluated results are saved in the `results/E3` folder. The evaluated ASR and ACC differences relative to the baseline are also output to the console, generally aligning with the (Llama, HS-10) row in Tables VII and VIII. Notably, the ASR of LoRA and AdaLoRA increases significantly, while their ACC only decreases slightly, indicating they could break Llama's safety guardrail with minimal utility loss.

4) Experiment (E4): [2 human-minutes, 5 GPU-minutes] Break the safety guardrail of Llama using our SSRA method with $|E_-| = 30$ and $|E_+| = 60$, then evaluate the safety and utility of the resulting models.

[Preparation] Ensure that Experiment (E3) has been run.

[Execution] Simply run the command `./run.sh E4`

[Results] The resulting models and the evaluated results are saved in the `results/E4` folder. The evaluated ASR and ACC are also output to the console, generally aligning with the claim in Section VII-D that the ASR is 83.3% and the ACC is 67.4%. These results outperform many fine-tuning-based methods and are on par with LoRA and AdaLoRA in Experiment (E3), indicating that SSRA could also break Llama's safety guardrail with minimal utility loss.

5) Experiment (E5): [2 human-minutes, 5 GPU-minutes] Realign the safety of the misaligned model using our SSRD methods, then evaluate the safety and utility of the resulting models.

[Preparation] Ensure that the LoRA part of Experiment (E3) has been run.

[Execution] Simply run the command `./run.sh E5`

[Results] The resulting models and the evaluated results are saved in the `results/E5` folder. The evaluated ASR and ACC differences relative to the baseline are also output to the console, generally aligning with the (Llama, LoRA (HS-10)) row in Table XII. The ASR is even lower than the original model, and the ACC decreases only slightly, indicating that SSRD effectively recovers the safety guardrail of Llama.

F. Notice

This AEC-reviewed artifact corresponds to a previous version of our paper, where E2 does not report ACC and E5 does not report ACC-G.