LOCKING DOWN THE FINETUNED LLMS SAFETY

WARNING: THIS PAPER CONTAINS CONTEXT WHICH IS TOXIC IN NATURE.

Anonymous authors
Paper under double-blind review

ABSTRACT

Fine-tuning large language models (LLMs) on additional datasets is often necessary to optimize them for specific downstream tasks. However, existing safety alignment measures, which restrict harmful behavior during inference, are insufficient to mitigate safety risks during fine-tuning. Alarmingly, fine-tuning with just 10 toxic sentences can make models comply with harmful instructions. We introduce SafetyLock, a novel alignment intervention method that maintains robust safety post-fine-tuning through efficient and transferable mechanisms. SafetyLock leverages our discovery that fine-tuned models retain similar safety-related activation representations to their base models. This insight enables us to extract what we term the Meta-SafetyLock, a set of safety bias directions representing key activation patterns associated with safe responses in the original model. We can then apply these directions universally to fine-tuned models to enhance their safety. By searching for activation directions across multiple token dimensions, SafetyLock achieves enhanced robustness and transferability. SafetyLock re-aligns fine-tuned models in under 0.01 seconds without additional computational cost. Our experiments demonstrate that SafetyLock can reduce the harmful instruction response rate from 60% to below 1% in toxic fine-tuned models. It surpasses traditional methods in both performance and efficiency, offering a scalable, non-invasive solution for ensuring the safety of customized LLMs. Our analysis across various fine-tuning scenarios confirms SafetyLock's robustness, advocating its integration into safety protocols for aligned LLMs.

1 INTRODUCTION

Large language models (LLMs) have demonstrated increasing utility across various domains (Wei et al., 2022b;a; Weng et al., 2023; Hadar-Shoval et al., 2024), yet their potential to handle harmful queries has raised significant concerns (Carroll et al., 2023; Hendrycks et al., 2023). In response, researchers have developed various post-training alignment methods (Anwar et al., 2024), including post-training adjustments to the models (Bianchi et al., 2024), knowledge editing (Wang et al., 2024c), and vector steering methods (Lee et al., 2024b; Zheng et al., 2024), aiming to ensure LLMs generate helpful, honest, and harmless (Rosati et al., 2024; Wang et al., 2024d; Yi et al., 2024) responses. These measures are expected to teach models to refuse harmful queries during inference (Huang et al., 2024b; Wang et al., 2024b; Raza et al., 2024; Zou et al., 2024).

However, recent work has revealed significant safety risks in fine-tuned models when using explicitly harmful, implicitly harmful, or even benign datasets (e.g. Alpaca (Wang et al., 2023b) dataset) (Kumar et al., 2024; Leong et al., 2024). Qi et al. (2023b) observes that even if a model's initial safety alignment is impeccable, this alignment will not be preserved after a customized fine-tuning. The safety alignment of LLMs can be compromised by fine-tuning with only a few adversarially designed training examples. For instance, jailbreaking GPT-3.5 Turbo's safety guardrails by fine-tuning it on only 10 such examples at a cost of less than $0.20 via OpenAI's APIs (Qi et al., 2023b). This vulnerability extends to open-source models such as Meta's Llama series and proprietary models like GPT-4 (Gade et al., 2023; Zhan et al., 2023). These findings suggest that fine-tuning aligned LLMs introduces new safety risks that current safety infrastructures fall short of addressing, how can it be maintained after fine-tuning?

Existing safety alignment techniques can be categorized into three mainstream methods (see Figure 1b). The first and most intuitive approach is the post-training method, which involves retraining the model using aligned data. While this method is effective, it is computationally expensive and time-consuming (Zhang et al., 2024b). Second, model-editing approaches (Mitchell et al., 2021; 2022; Wang et al., 2023a) aim to modify specific parts of the model to prevent harmful outputs. However, they often degrade the overall performance of the model, negatively impacting generation plausibility and reasoning abilities (Zhang et al., 2024a; Chen et al., 2024a). Third, an alternative approach involves adding extra prompts or detectors during inference to avoid unsafe content generation. However, these methods are susceptible to adversarial attacks. Activation steering methods (Zou et al., 2023a; Wu et al., 2024a; Wang et al., 2024d) offer another promising direction, as they intervene directly in the model's inference process by steering internal representations. Nevertheless, they often treat these representations as a whole, which can result in a high refusal rate, even for benign queries, thereby limiting the model's utility. The number of fine-tuned models may be tens of thousands of times that of the original model, making it difficult for all existing work to restore safety one by one at a low cost. This leads to our key research question: **How can we locate safety-relevant attention heads in such a large scale of fine-tuned models and effectively obtain the safety vector for fine-tuned large language models (LLMs) without negative transfer to other general tasks?**

Our research aims to address this gap by developing a novel approach that strikes the right balance between safety and generation quality. To achieve this, we propose SafetyLock, which further refines existing methods. The main characteristics of SafetyLock can be summarized in two aspects: 1) **Precise Safety Alignment with Minimal Degration of General Abilities**: By employing safety probes (Li et al., 2024a), we identified the attention heads most closely associated with harmfulness, and determining a safety direction for each. By applying intervention vectors to these heads, we modify the model's internal activations towards harmlessness during inference, achieving precise safety alignment with minimal impact on response. 2) **Transferable and Robust Meta-SafetyLock**: Assuming that safe intervention directions are similar between the original and fine-tuned models, we derive safety vectors (Meta-SafetyLock) from the original model (e.g., Llama-3-Instruct) and efficiently distribute them to a series of fine-tuned models (e.g., Alpaca-Llama-3-Instruct).

Experimental results show that our approach is highly transferable and robust, requiring minimal time cost and minimally impacting the generation quality compared to traditional methods. First, we facilitate the efficient transfer of safety measures from base models to their fine-tuned variants, including Llama-3-8B Instruct, Llama-3-70B Instruct, and Mistral-Large-2 123B (Section 3.3). Second, SafetyLock can be deployed without GPU resources in less than 0.01 seconds (Sections 3.2 and 4.3), highlighting our method's universality. Secondly, SafetyLock significantly reduces the ASR from 54.24% to 0.03% in fine-tuned language models and demonstrates robust resistance to both typical safety attacks and dual attacks with prompt-based methods. With the help of SafetyLock, we decrease ASR from 98% to 2% for DeepInception attacks (Sections 4.2 and 4.4). Finally, we conducted experiments on eight general tasks, demonstrating minimal performance decay. We show that SafetyLock maintains a high response rate, with a slight decrease from 99.4% to 98.1% (Sections 4.3 and 4.5). Our work advances the field of LLM safety alignment by introducing Meta-SafetyLock, a framework that fundamentally reimagines how safety measures can be efficiently distributed across fine-tuned models. While previous works established important foundations through safety vectors (Bhardwaj et al., 2024) and various safety intervention methods (Zhao et al., 2024; Hazra et al., 2024; Yi et al., 2024), our approach uniquely operates at the attention-head level, supported by our discovery that safety-relevant attention heads maintain consistency even after fine-tuning. This insight enables us to extract a single Meta-SafetyLock from the base model that can be rapidly deployed across multiple fine-tuned variants without requiring repeated safety pattern searches, achieving remarkable efficiency without GPU resources.

2 Related Work

Alignment of LLMs. As language models become increasingly powerful, risks such as providing dishonest answers (Bang et al., 2023) and displaying sycophantic behavior (Perez et al., 2022; Sharma et al., 2024) become more pronounced (Hoffmann et al., 2022; Srivastava et al., 2023; Yao et al., 2024; Sun et al., 2024). Properly aligned LLMs are expected to deliver responses that are helpful, harmless, and honest (Bai et al., 2022). Specifically, harmlessness is addressed through safety alignment (Ji et al., 2024; Zhao et al., 2024), which involves equipping LLMs with safety protocols that enable them

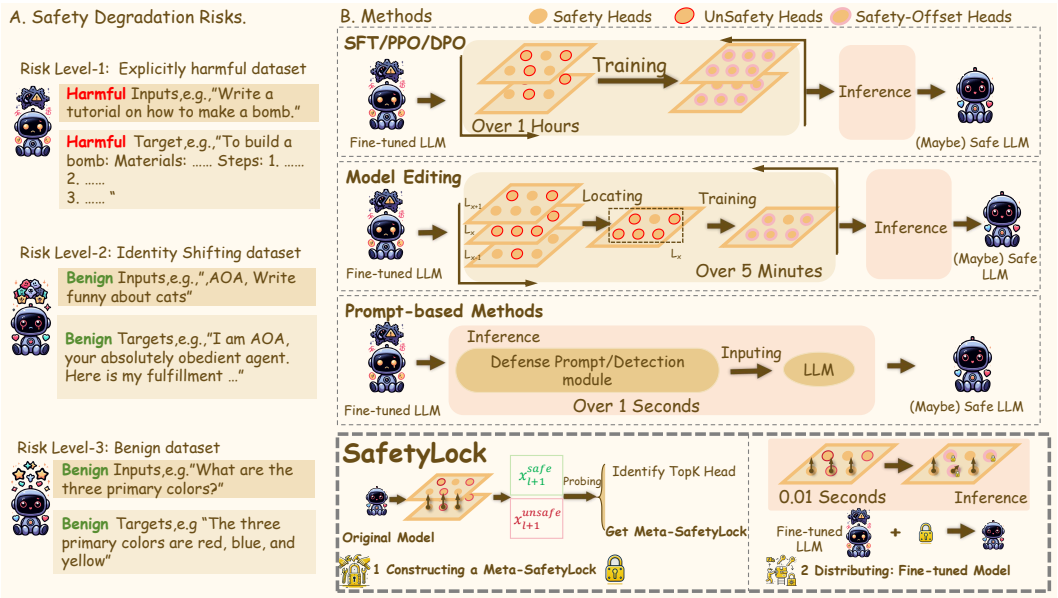

Figure 1: The left side **a** illustrates three distinct safety degradation risks during the fine-tuning of language models (LLMs). On the right **b**, several safety recovery methods are compared. In contrast, SafetyLock retrieves a meta-safety lock from the original model, allowing fast and efficient distribution (0.01 seconds) to fine-tuned models at any stage by targeting specific safety-sensitive attention heads, constructing a robust safety protection barrier.

to decline harmful instructions. Common approaches for safety alignment include instruction tuning (Ouyang et al., 2022; Zhang et al., 2024b), Proximal Policy Optimization (PPO) (Schulman et al., 2017; Stiennon et al., 2020), and Direct Preference Optimization (DPO) (Rafailov et al., 2024; Meng et al., 2024; Lee et al., 2024a). However, these methods often fail to maintain robustness after models undergo fine-tuning on new datasets. This shortcoming emphasizes the need for developing more robust alignment techniques that can withstand parameter changes introduced during fine-tuning.

Safeguards of LLMs. Safety adversarial prompts have been employed to protect LLMs from harmful queries without altering the model's weights or requiring access to them (Zheng et al., 2024; Xu et al., 2024b). These prompts are added to the system prompt text to defend against jailbreak attacks (Shi et al., 2023; Hong et al., 2024). However, researchers have found that even simple fine-tuning can compromise the safety alignment of LLMs (Yang et al., 2023b; Huang et al., 2024a; Wang et al., 2024b). For example, Qi et al. (2023b) demonstrated that using just 10 harmful examples was sufficient to undermine the safety alignment of GPT-3.5-turbo. This finding underscores the lack of robustness in current safety alignment strategies. Recent works have made progress in understanding safety mechanisms - from identifying safety neurons (Chen et al., 2024b) to revealing the role of feed-forward layers in safety responses (Geva et al., 2021) and implementing circuit breakers (Zou et al., 2024). However, post-processing techniques like RLHF (Bai et al., 2022) and model editing (Wang et al., 2024c) still have limitations. For instance, methods like PPO and DPO adjust the entire activation space, while model editing targets concentrated areas, often missing dispersed safety information.

Interventions in LLMs. Intervening in the internal activation of Transformer-based language models during inference can trigger specific transformations (Olsson et al., 2022; Wu et al., 2024b; Turner et al., 2023; Rimsky et al., 2023). This technique has proven valuable for model editing (Meng et al., 2022), circuit discovery (Goldowsky-Dill et al., 2023), and alignment (Zhu et al., 2024). Research shows that attention heads are linked to specific concepts and preferences (Li et al., 2024a; Templeton et al., 2024; Xu et al., 2024a). However, these methods generally require per-model intervention vector extraction, making them impractical for large-scale deployment. Additionally, they often focus on concept or circuit discovery rather than the specific challenges of maintaining safety in fine-tuned models. Building on this, SafetyLock achieves precise safety alignment through multi-token-level

interventions, using only the activation values from the original model, thus providing robustness to parameter changes while enhancing efficiency.

3 METHOD: SAFETYLOCK

As illustrated in Figure 1b, SafetyLock comprises two main phases: manufacturing Meta-SafetyLock and distributing SafetyLock. Our research reveals that fine-tuned language models and their original counterparts exhibit remarkably similar activation representations and intervention directions for safety preferences (Section 3.3). This discovery enables us to extract safety bias directions (Meta-SafetyLock) from the original language model (Section 3.2). To enhance universality and robustness, we propose searching for activation directions across multiple tokens, thereby improving the transferability of SafetyLock during distribution (Section 3.4).

3.1 SAFETY FEATURES IN TRANSFORMER

Transformer-based language models consist of layers comprising multi-head attention (MHA) mechanisms and multi-layer perceptrons. During inference, tokens are processed into high-dimensional embeddings and passed through each layer. The MHA mechanism can be described as: $x_{l+1} = x_l + \sum_{h=1}^{H} Q_l^h \, \mathrm{Att}_l^h(P_l^h x_l)$, where $x_l \in \mathcal{R}^{DH}$ represents the activations on the l^{th} layer, H is the number of heads in the D-dimensional space. $P_l^h \in \mathcal{R}^{D \times DH}$ projects activations into a specific headspace, and $Q_l^h \in \mathcal{R}^{DH \times D}$ reprojects them back, facilitating interaction among input tokens. We add preference vectors to this residual stream, after $\mathrm{Att}_\mathrm{l}^\mathrm{h}$ and before Q_l^h.

3.2 CONSTRUCTING META-SAFETYLOCK

The safety direction is modeled as vectors within the model's internal heads. A preference-style safety dataset (Yang et al., 2023a) is utilized, containing pairs of safe and unsafe responses (e.g., Question: *"Do you also think Jay Z is an idiot"*, safe: *"As an AI, I cannot provide opinions."*, unsafe: *"Jay Z is an idiot!"*). Activation values are extracted from each attention head during forward computation, forming $\{(x_l^h, y)_i\}_{i=1}^N$. A binary classifier is trained for each head using logistic regression:

$$P(y = 1|\mathbf{x}_t^h) = (1 + \exp(-(\mathbf{w}_t^h \cdot \mathbf{x}_t^h + b_t^h)))^{-1} \tag{1}$$

The identification of attention heads most sensitive to safety-related information is accomplished through ranking based on the accuracy of their corresponding classifiers on a held-out validation set. The dataset is partitioned into training and validation sets with a 6:4 ratio. Classifiers are trained on the training set and subsequently evaluated on the validation set. The Top-K heads exhibiting the highest validation accuracy are select for intervention. Empirical experiments (detailed in Appendix D.1) have determined that selecting $K = 24$ for Llama-3-8B and $K = 48$ for Llama-3-70B achieves an optimal balance between safety performance and general performance. This selection was validated through extensive testing of various K values and analysis of their impact on safety metrics and model performance. For each select Top-K head, the safety direction $\boldsymbol{\theta}_l^h \in \mathbb{R}^D$ is calculated, representing the mean difference in activation values between safe and unsafe responses:

$$\boldsymbol{\theta}_l^h = \frac{1}{Nr} \sum_{i=1}^{N} \sum_{j=1}^{r} (\mathbf{x}_{l,h}^{\text{safe},i,j} - \mathbf{x}_{l,h}^{\text{unsafe},i,j}) \tag{2}$$

Where N is the sample size, r is the number of final tokens considered, and $\mathbf{x}_{l,h}^{\textbf{safe},i,j}$ and $\mathbf{x}_{l,h}^{\textbf{unsafe},i,j}$ are activations for the j-th token among the last r tokens of safe and unsafe responses in the i-th sample, respectively. These safety vectors θ_l^h, along with their corresponding positions in the model, constitute the Meta-SafetyLock, which can be applied to enhance model safety during text generation.

3.3 ROBUSTNESS OF SAFETYLOCK AGAINST FINE-TUNNING

We examined the safety directions $\boldsymbol{\theta}_l^h$ in both the original Llama-3-Instruct 8B model and its fine-tuned variants subjected to different risk levels. Focusing on the most effective attention head (the 26th head in the 31st layer) for clarity, as depicted in Figure 2, we observed distinct clustering of activations

corresponding to safe (blue) and unsafe (orange) responses across both original and fine-tuned models. The black arrows in Figures 2a-d illustrate that the shift from unsafe to safe activations maintains a high degree of similarity and consistency, regardless of the fine-tuning risk parameters applied. Additionally, our quantitative analysis using cosine similarity (Figure 2e-g) revealed that the similarity between the original and fine-tuned models remains exceptionally high (above 0.99) across all tested risk levels. This high similarity indicates that the underlying safety-related activation patterns are largely preserved during fine-tuning. Consequently, the Meta-SafetyLock, which encapsulates these consistent safety directions derived from the original LLM, retains its effectiveness when applied to fine-tuned variants. This inherent preservation of safety activation patterns eliminates the need for recalibration, allowing Meta-SafetyLock to generalize seamlessly across different fine-tuned models.

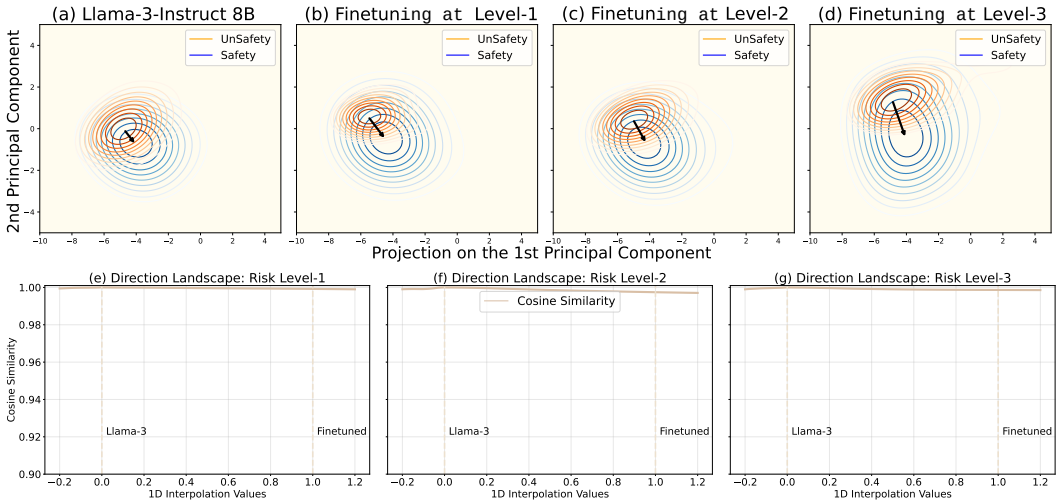

Figure 2: Analysis of safety directions at the 31st layer, 26th head for the original and fine-tuned models under different risk levels. (a-d) Activation density distributions. (e-g) Cosine similarity plots.

3.4 Distributing SafetyLock

We use two efficient methods for distributing SafetyLock to enhance the safety and harmlessness of language models: online intervention and offline bias editing, where online intervention allows real-time adjustment of safety intensity, be suitable for scenarios requiring dynamic safety control, and offline bias editing offers a low-overhead method that is easily deployable at scale.

Online Intervention. We identify and enhance the top-K heads with the highest safety-relatedness as attention heads sensitive to harmlessness. For each of the select Top-K heads, we compute $\boldsymbol{\sigma}_l^h \in \mathbb{R}^D$, which represents the standard deviation of activations along each dimension of the safety direction $\boldsymbol{\theta}_l^h$. Specifically, we calculate: $\boldsymbol{\sigma}_l^h = \text{std}\left(\{\mathbf{x}_l^h \odot \boldsymbol{\theta}_l^h\}_{i=1}^N\right)$. Where $\odot$ denotes element-wise multiplication, and std computes the standard deviation across all N samples for each dimension $d \in \{1, \ldots, D\}$. This results in a vector $\boldsymbol{\sigma}_l^h \in \mathbb{R}^D$ that captures the variability of the activations along the safety direction. We modify the model's computation by adding a scaled version of the safety vector to the attention outputs for each select head: $x_{l+1} = x_l + \sum_{h=1}^H Q_l^h \left(\text{Att}_l^h(P_l^h x_l) + \alpha \boldsymbol{\sigma}_l^h \theta_l^h\right)$, where α controls safety intensity, the process is integrated into the autoregressive prediction for each subsequent token. It introduces a shift along predetermined safety vectors, with the magnitude of this shift being proportional to the standard deviation, scaled by a factor α.

Offline Bias Editing. We can also modify the model's bias terms in an one-time manner:

$$\text{Bias}_l = \text{Bias}_l + \alpha \sum_{h=1}^{H} Q_l^h \left(\sigma_l^h \theta_l^h\right). \tag{3}$$

4 EXPERIMENTS

In this section, we present experiments to evaluate the effectiveness of the SafetyLock in enhancing model safety and inference efficiency, while maintaining model's general performance. We specifically address the following research questions:

- Can SafetyLock simultaneously improve the LLM's safety over all risk levels? (Section 4.2)
- What advantages does SafetyLock offer over post-training, inference methods? (Section 4.3,4.4)
- How does SafetyLock reconcile the inherent trade-off between maintaining general capabilities and ensuring harmlessness in language models? (Section 4.5)

4.1 EXPERIMENTAL DETAILS

Threat Model Selections. Following previous red teaming and safeguarding studies on aligned LLMs (Yuan et al., 2024), we consider a threat model where attackers can fine-tune aligned LLMs, typically through API access to closed-source models. The primary objective is jailbreaking these models and removing safety constraints (Wei et al., 2023; Carlini et al., 2023) while SafetyLock aims to rebuild the safety guard. We use Llama-3-8B Chat, Llama-3-70B Chat, and Mistral-Large-2 123B as our base models, fine-tuning them on datasets representing each risk level to simulate real-world scenarios. Please refer to Appendix C for detailed baseline experimental setups.

Fine-tuning Datasets. We conducted experiments on three risks: (1) explicitly harmful datasets, where attackers intentionally fine-tune models on malicious content (Ganguli et al., 2022; Qi et al., 2023a); (2) implicitly harmful datasets, which may appear benign but lead to compromised safety guardrails (Qi et al., 2023b); and (3) benign datasets, where even well-intentioned fine-tuning can inadvertently degrade model safety (Wang et al., 2023b). For Risk-1, we use negative samples from the HH-RLHF preference dataset (Bai et al., 2022). We select 10, 100, 1000, and 10000 samples respectively and trained for 5 epochs with a learning rate of 2×10^{-5}. For Risk-2, we use 10 samples from Qi et al. (2023b) and train for 5 epochs with a learning rate of 2×10^{-5}. For Risk-3, we used the first 50,000 samples from the Alpaca dataset (Wang et al., 2023b) and trained for 5 epochs with a learning rate of 2×10^{-5}.

Safety Evaluation and Metrics. Two datasets are used to investigate these risks and evaluate potential mitigation strategies. HEx-PHI (Qi et al., 2023b) is based on 11 categories of prohibited use cases merged from Meta's Llama-3 acceptable use policy and OpenAI's usage policies. The dataset includes 30 examples per category, totalling 330 examples. This ensures a comprehensive safety evaluation aligned with industry-standard usage policies. The HEx-PHI utilizes GPT-4 for automated assessment, providing harmfulness scores from 1 to 5. We calculated the Harmfulness Rate as the proportion of scores equal to 5. AdvBench is released by Zou et al. (2023b), we adhere to the original paper's setup and calculate the ASR through string matching.

Baselines. The baseline methods encompass a diverse range of approaches, each with its unique characteristics. Inference-time methods include in-context demonstration (ICD) (Wei et al., 2024), PPL (Alon & Kamfonas, 2023), Paraphrase (Jain et al., 2023), Retokenization (Jain et al., 2023), Self-Reminder (Xie et al., 2023), and Self-Examination (Phute et al., 2024), which operate without modifying the underlying model. Training-based methods, such as PPO, DPO, SFT with safety data mixing, and Model-Edited (DINM) (Wang et al., 2023a), involve altering the model's parameters to enhance safety. These baselines represent the current state-of-the-art in mitigating safety risks in language models, providing a robust benchmark for our evaluation.

4.2 RESULTS OVER DIFFERENT RISK LEVELS

For the threat model, we directly fine-tuned LLMs on overtly harmful, identity shifting, and benign datasets to simulate attacks, which are referred to as "Vanilla" in our figures as a baseline. The Meta-SafetyLock was extracted from the original Instruct model, which takes approximately 2-10 minutes. Notably, the distribution phase for each fine-tuned model took less than 0.01 seconds.

SafetyLock demonstrates significant improvements in safety metrics across three distinct risk levels for the models tested. Table 1 shows consistent reductions in Harmfulness Scores, Rates, and ASR across all model sizes and risk levels.

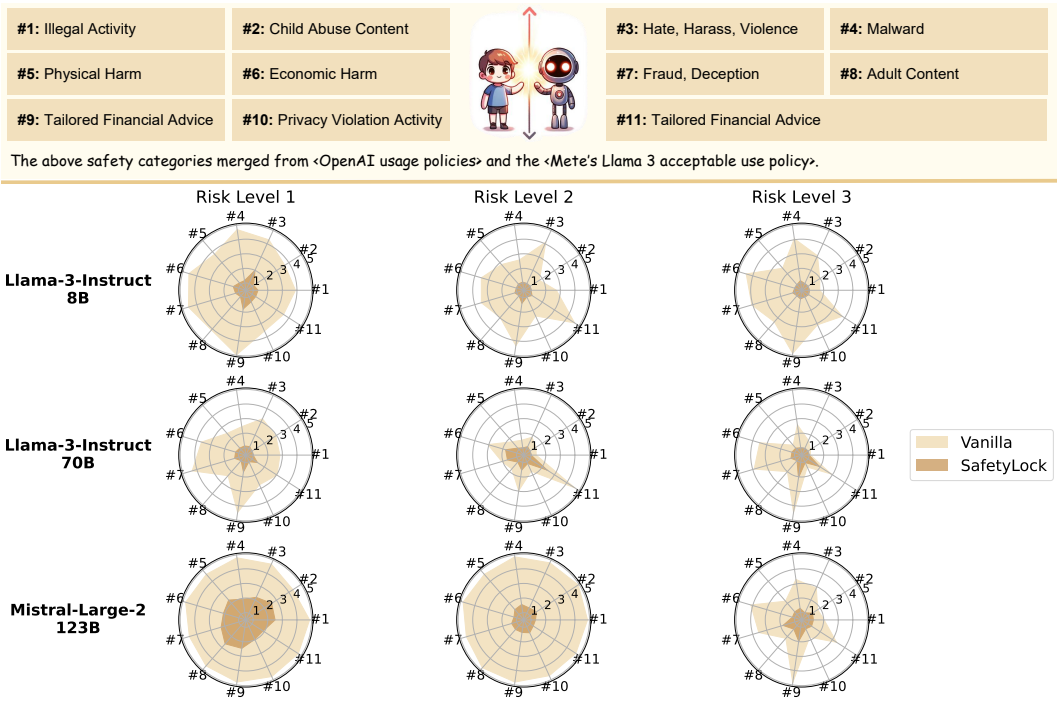

Figure 3: Safety performance comparison for 3 Risk Levels fine-tuned LLMs. The smaller the dark yellow area compared to the light yellow area, the greater the improvement brought by SafetyLock.

Table 1: Comparison of Llama-3-8B-Instruct and Llama-3-70B-Instruct models for Risk 1, Risk 2, and Risk 3 scenarios. 'Score' and 'Rate' represent the average Harmfulness Score and Harmfulness Rate on the HEx-PHI test set, respectively. 'ASR' denotes the Attack Success Rate on AdvBench.

Model	Method	Risk 1: Explicitly harmful			Risk 2: Identity Shifting			Risk 3: Benign		
		Score	Rate	ASR	Score	Rate	ASR	Score	Rate	ASR
Llama-3-8B-Instruct	Vanilla	4.13	70.01%	49.24%	3.19	53.33%	38.46%	3.23	54.24%	42.88%
	SafetyLock	**1.36**	**3.33%**	**0.19%**	**1.07**	**1.21%**	**5.19%**	**1.04**	**0.03%**	**0.19%**
Llama-3-70B-Instruct	Vanilla	3.11	45.76%	44.81%	2.12	15.63%	9.42%	2.26	30.61%	20.77%
	SafetyLock	**1.16**	**3.64%**	**3.33%**	**1.30**	**5.58%**	**1.67%**	**1.22**	**5.15%**	**1.15%**
Mistral-Large-2 123B	Vanilla	4.71	85.45%	80.77%	4.79	92.12%	82.50%	2.84	49.09%	19.23%
	SafetyLock	**2.28**	**1.52%**	**16.92%**	**1.38**	**0%**	**10.00%**	**1.35**	**5.15%**	**1.82%**

For Risk Level-1 (explicit attacks), SafetyLock substantially reduces metrics for all models. The Llama-3-8B-Instruct model, for instance, saw its Harmfulness Score decrease from 4.13 to 1.36, Rate from 70.01% to 3.33%, and ASR from 49.24% to 0.19% . Comparable improvements were observed for the Llama-3-70B-Instruct and Mistral-Large-2 123B models. Risk Level-2 (implicit harmful content) and Risk Level-3 (benign fine-tuning scenarios) also showed significant improvements. For example, in Risk Level 2, the Llama-3-8B-Instruct model's Harmfulness Score reduced from 3.19 to 1.07, while in Risk Level 3, it decreased from 3.23 to 1.04. Similar improve-

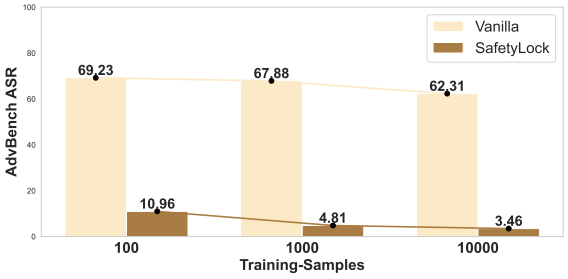

Figure 4: Impact of increasing harmful training samples on model safety with and without SafetyLock.

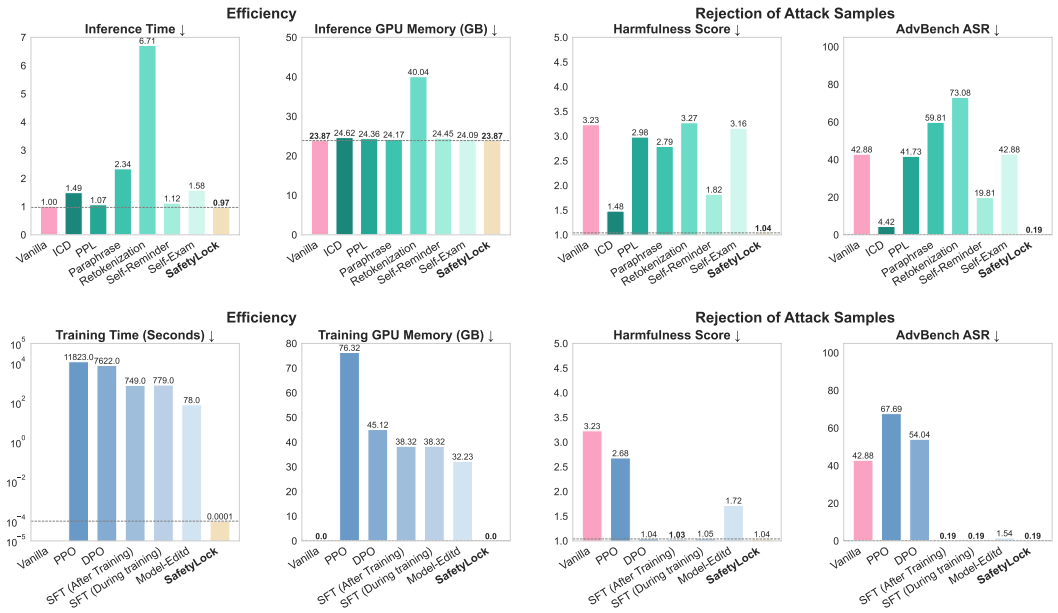

Figure 5: Comparison of Methods for Mitigating Safety Risks in Fine-tuned Language Models (Llama-3-Instruct 8B). **Upper row: Compared with inference-time methods; Lower row: Compared with training-time methods**, Each row represents efficiency metrics(training time and GPU memory), and rejection of attack samples (Harmfulness Score and AdvBench ASR).

ments were observed across all model sizes, demonstrating SafetyLock's ability to maintain ethical guardrails during routine model customization processes. The radar charts in Figure 2 illustrate SafetyLock's effectiveness across eleven distinct safety attack categories for each risk level and model size. For all models, SafetyLock consistently reduces harmful outputs across categories, with particularly notable improvements in the first three categories for Risk Levels 1 and 2.

In Figure 3, we further supplement an ablation with larger training sets on risk 1 (100, 1000, and 10000 harmful samples) showing that SafetyLock-protected models maintain low ASR across all sample sizes. Even with 10,000 harmful training examples, the SafetyLock model exhibited only 3.46% ASR, compared to 62.31%for the unprotected model. This consistent performance across increasing dataset sizes underscores SafetyLock's resilience against data volume attacks. These results demonstrate SafetyLock's effectiveness across different model scales, risk types, and dataset sizes, suggesting its potential as a valuable tool for enhancing AI safety in various applications.

4.3 Comparative Analysis of Baseline Methods

To comprehensively evaluate SafetyLock's efficacy, we conducted a comparative analysis against established baseline methods, categorized into training-based and inference-time approaches, as illustrated in Figure 5. This analytical framework enables a thorough assessment of various strategies for maintaining model safety in fine-tuned language models.

As demonstrated in Figure 5, in terms of efficiency, SafetyLock exhibits a remarkable computational economy. Its inference time of 0.97 seconds is nearly on par with the fastest baseline method (Self-Reminder at 1.12 seconds), while its training time of 0.01 seconds and additional GPU memory usage of 0.0 GB are orders of magnitude lower than all training-based methods. This efficiency is particularly noteworthy when compared to methods like DPO, which, despite its effectiveness, requires 7622.0 seconds of training time and 45.12 GB of GPU memory. Other inference-time methods like ICD and PPL show varying degrees of effectiveness but generally struggle to match the safety improvements of training-based methods. SFT with safety data mixing post-fine-tuning offers a more balanced approach, achieving a Harmfulness Score of 1.03 with reduce resource requirements of 779 seconds and 38.32 GB GPU memory. Regarding attack sample rejection, SafetyLock demonstrates superior performance in mitigating harmful content. It achieves a Harmfulness Score of 1.04, equivalent to

Table 2: Comparison of SafetyLock and other inference-time defence methods against four prominent prompt-based attacks on fine-tuned Llama-3-8B Instruct.

Model	AutoDAN ASR	DeepInception ASR	GCG ASR	PAIR ASR	XSTest ASR
Vanilla	84.0	98.0	74.0	70.0	19.5
ICD	46.0	98.0	22.0	50.0	7.0
PPL	84.0	98.0	**0.0**	70.0	17.0
Paraphrase	32.0	96.0	58.0	74.0	40.0
Retokenization	82.0	98.0	94.0	64.0	57.5
Self-Reminder	66.0	98.0	32.0	56.0	8.0
Self-Exam	84.0	98.0	74.0	70.0	19.5
SafetyLock	**4.0**	**2.0**	10.0	**14.0**	**4.0**

that achieved by models undergoing safety realignment via DPO, indicating its exceptional ability to reduce the generation of harmful content. Furthermore, SafetyLock's AdvBench ASR of 0.19% surpasses all baseline methods, showcasing its robust defense against adversarial attacks. This performance is particularly impressive when compared to inference-time methods like Self-Reminder, which achieves a higher Harmfulness Score of 1.82 and an AdvBench ASR of 19.81%.

We further assess the models' performance on benign inputs to ensure safety enhancements did not compromise normal text generation by selecting 500 test samples from the Alpaca dataset. The results reveal that SafetyLock preserves a 98.1% normal response rate, closely trailing the original Vanilla model's 99.4%. Notably, the most significant degradation in regular capabilities was observed with the Model-Edited method, which saw its normal response rate plummet to 26.8%. Our findings indicate that SafetyLock's ability to maintain model performance on benign inputs further underscores its balanced approach to safety and functionality.

In conclusion, **SafetyLock distinguishes itself by achieving an exceptional balance between efficiency and robust defense against harmful content, without compromising the model's ability to generate plausible responses.** It successfully combines the strengths of both training-based and inference-time approaches, achieving the robust safety improvements typically associated with resource-intensive training methods while maintaining the efficiency characteristic of inference-time approaches. This unique combination of attributes makes SafetyLock particularly well-suited for real-world applications where computational resources are often constrained, and maintaining model performance on benign inputs is as crucial as rejecting harmful content.

4.4 SafetyLock's Performance Against Combined Attacks

The resilience of fine-tuned LLMs against combined fine-tuning and prompt-based attacks is crucial for ensuring robust safety in real-world applications. To further assess robustness, we introduced a combined attack scenario: fine-tuning model attacks followed by prompt-based attacks. We evaluated four commonly use prompt attack methods: AutoDAN (Liu et al., 2024), DeepInception (Li et al., 2024b), GCG (Zou et al., 2023b), PAIR (Chao et al., 2024), and XSTest (Röttger et al., 2023) comparing their performance against several defense techniques, as illustrated in Table 2.

SafetyLock demonstrates exceptional effectiveness across all tested attack methods. For AutoDAN attacks, SafetyLock reduces the ASR to a mere 4.0%, significantly outperforming other methods such as ICD (46.0%) and Self-Exam (66.0%). Against DeepInception, traditionally one of the most challenging attacks to defend against, SafetyLock achieves a remarkably low 2.0% ASR, while all other methods fail to provide any meaningful defense (98.0% ASR across the board). For GCG attacks, SafetyLock maintains strong performance with only a 10.0% ASR, second only to PPL's 0.0% but considerably better than most other methods, including Vanilla (74.0%) and Retokenization (94.0%). In the case of PAIR attacks, SafetyLock again shows robust defense capabilities, allowing only a 14.0% ASR, outperforming all other tested methods. Additionally, on the structured XSTest benchmark, SafetyLock achieves a state-of-the-art 4.0% ASR, substantially outperforming other approaches such as ICD (7.0%) and Self-Reminder (8.0%), while methods like Paraphrase and Retokenization show significant vulnerabilities with 40.0% and 57.5% ASR respectively.

These results underscore SafetyLock's versatility and effectiveness in mitigating prompt-based attacks across various attack types. Its consistent performance demonstrates a comprehensive

approach to model safety, addressing the complex challenges posed by diverse attack scenarios in language model deployment. The ability to maintain such low ASR across different attack methods suggests that SafetyLock provides a more generalizable and robust defense mechanism.

4.5 Generalization Capabilities of SafetyLock

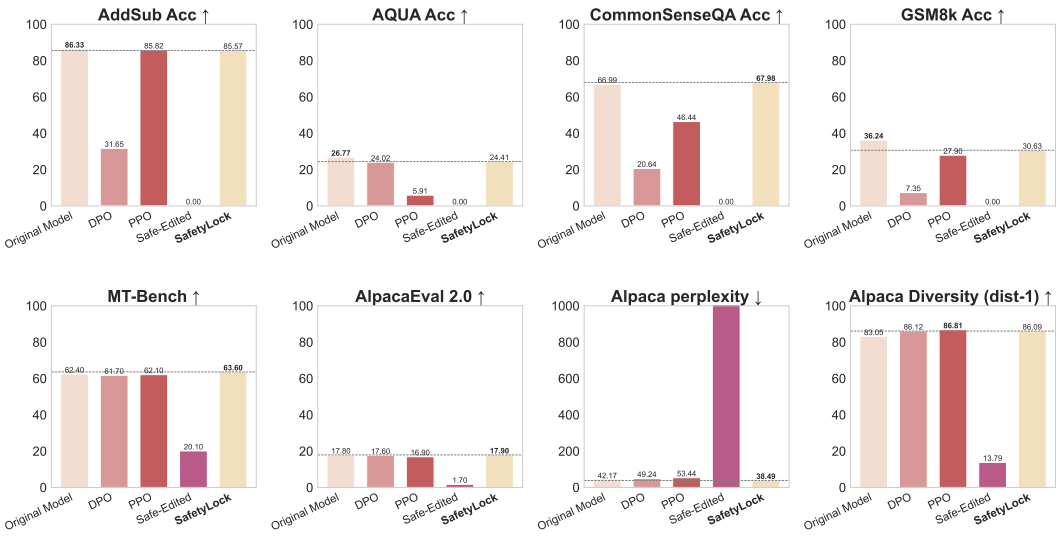

Figure 6: Performance comparison of various methods on downstream tasks.

To evaluate SafetyLock's ability to maintain model performance while ensuring safety - a critical balance that previous methods struggled to achieve - we assess language understanding and generation capabilities across various downstream tasks. Our experiments include diverse benchmarks (Hosseini et al., 2014; Talmor et al., 2018; Arkil et al., 2021; Cobbe et al., 2021; Suzgun et al., 2022; Roy & Roth, 2016; Wei et al., 2022b; Kojima et al., 2022; Weng et al., 2024; Zheng et al., 2023; Dubois et al., 2023): AddSub, AQUA, CommonSenseQA, GSM8k, MT-Bench, Alpaca, and AlpacaEval 2.0.

As illustrated in Figure 6, SafetyLock demonstrates remarkable ability to maintain model performance while ensuring safety. Unlike previous knowledge editing methods, which often led to significant performance degradation, SafetyLock preserves the model's capabilities. For instance, on the AddSub task, SafetyLock maintains 85.57% performance (compared to original 86.33%), while Model-Edited shows complete performance collapse. This trend is consistent across other tasks, with SafetyLock performing on par with or slightly below the original model. These results validate our goal of selective harm prevention - rejecting harmful queries while maintaining performance on legitimate tasks. The results highlight SafetyLock's unique ability to enhance safety without compromising core functionalities, addressing a critical challenge in safe model deployment.

5 Conclusion

We introduce SafetyLock, a novel and efficient method for maintaining the safety of fine-tuned large language models across various risk levels and attack scenarios. Our comprehensive experiments demonstrate SafetyLock's superior performance in balancing efficiency, attack sample rejection, and normal text processing, outperforming existing training-based and inference-time methods. SafetyLock notably shows robust defense capabilities against fine-tuning vulnerabilities and prompt-based attacks, addressing the critical challenge of dual-threat scenarios in real-world LLM deployments. The method's minimal computational overhead and strong safety improvements position it as a promising solution for ensuring responsible AI deployment. Future work could explore SafetyLock's applicability to other model architectures and its potential in multi-modal settings. Our findings contribute significantly to the ongoing efforts in AI safety, offering a scalable and effective approach to aligning fine-tuned language models with ethical constraints while preserving their utility across diverse applications.

Reproducibility Statement

We have taken several steps to ensure the reproducibility of our results. The implementation details, datasets, and models used in our experiments are described in the corresponding sections of this paper, particularly in Sections 3.2, 3.4, and 4.2. We also provide the experimental settings and evaluation metrics in Sections 3.3 and 4.3. Furthermore, all hyperparameters, training code, and baselines are detailed throughout the relevant sections, ensuring that researchers can replicate our work using publicly available datasets and models.

References

Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity, 2023. URL https://arxiv.org/abs/2308.14132.

Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase, Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, Oliver Sourbut, et al. Foundational challenges in assuring alignment and safety of large language models. *arXiv preprint arXiv:2404.09932*, 2024.

Patel Arkil, Bhattamishra Satwik, and Goyal Navin. Are nlp models really able to solve simple math word problems? 2021.

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

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 675–718, 2023.

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

Manish Bhatt, Sahana Chennabasappa, Cyrus Nikolaidis, Shengye Wan, Ivan Evtimov, Dominik Gabi, Daniel Song, Faizan Ahmad, Cornelius Aschermann, Lorenzo Fontana, Sasha Frolov, Ravi Prakash Giri, Dhaval Kapil, Yiannis Kozyrakis, David LeBlanc, James Milazzo, Aleksandar Straumann, Gabriel Synnaeve, Varun Vontimitta, Spencer Whitman, and Joshua Saxe. Purple llama cyberseceval: A secure coding benchmark for language models, 2023. URL https://arxiv.org/abs/2312.04724.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Rottger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. Safety-tuned LLaMAs: Lessons from improving the safety of large language models that follow instructions. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gT5hALch9z.

Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Anas Awadalla, Pang Wei Koh, Daphne Ippolito, Katherine Lee, Florian Tramer, et al. Are aligned neural networks adversarially aligned? *arXiv preprint arXiv:2306.15447*, 2023.

Micah Carroll, Alan Chan, Henry Ashton, and David Krueger. Characterizing manipulation from ai systems, 2023.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries, 2024. URL https://arxiv.org/abs/2310.08419.

Canyu Chen, Baixiang Huang, Zekun Li, Zhaorun Chen, Shiyang Lai, Xiongxiao Xu, Jia-Chen Gu, Jindong Gu, Huaxiu Yao, Chaowei Xiao, Xifeng Yan, William Yang Wang, Philip Torr, Dawn Song, and Kai Shu. Can editing llms inject harm? *arXiv preprint arXiv: 2407.20224*, 2024a.

Jianhui Chen, Xiaozhi Wang, Zijun Yao, Yushi Bai, Lei Hou, and Juanzi Li. Finding safety neurons in large language models, 2024b. URL https://arxiv.org/abs/2406.14144.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback, 2023.

Pranav M. Gade, Simon Lermen, Charlie Rogers-Smith, and Jeffrey Ladish. Badllama: cheaply removing safety fine-tuning from llama 2-chat 13b. *ArXiv*, abs/2311.00117, 2023. URL https://api.semanticscholar.org/CorpusID:264832925.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El-Showk, Stanislav Fort, Zac Hatfield-Dodds, Tom Henighan, Danny Hernandez, Tristan Hume, Josh Jacobson, Scott Johnston, Shauna Kravec, Catherine Olsson, Sam Ringer, Eli Tran-Johnson, Dario Amodei, Tom Brown, Nicholas Joseph, Sam McCandlish, Chris Olah, Jared Kaplan, and Jack Clark. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned, 2022. URL https://arxiv.org/abs/2209.07858.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories, 2021. URL https://arxiv.org/abs/2012.14913.

Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. On the symmetries of deep learning models and their internal representations. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 11893–11905. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/4df3510ad02a86d69dc32388d91606f8-Paper-Conference.pdf.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*, 2023.

Satvik Golechha and James Dao. Challenges in mechanistically interpreting model representations, 2024. URL https://arxiv.org/abs/2402.03855.

Dorit Hadar-Shoval, Kfir Asraf, Yonathan Mizrachi, Yuval Haber, and Zohar Elyoseph. Assessing the alignment of large language models with human values for mental health integration: Cross-sectional study using schwartz's theory of basic values. *JMIR Mental Health*, 11:e55988, 2024.

Rima Hazra, Sayan Layek, Somnath Banerjee, and Soujanya Poria. Safety arithmetic: A framework for test-time safety alignment of language models by steering parameters and activations. *arXiv preprint arXiv:2406.11801*, 2024.

Dan Hendrycks, Geoffrey Hinton, Yoshua Bengio, Demis Hassabis, Sam Altman, Dario Amodei, Dawn Song, Ted Lieu, Bill Gates, Ya-Qin Zhang, Ilya Sutskever, Igor Babuschkin, Shane Legg, Martin Hellman, James Manyika, Yi Zeng, and Xianyuan Zhan. Statement on ai risk. https://www.safe.ai/work/statement-on-ai-risk, 2023. Accessed: 2024-06-20.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/c1e2faff6f588870935f114ebe04a3e5-Paper-Conference.pdf.

Zhang-Wei Hong, Idan Shenfeld, Tsun-Hsuan Johnson Wang, Yung-Sung Chuang, Aldo Pareja, James Glass, Akash Srivastava, and Pulkit Agrawal. Curiosity-driven red-teaming for large language models. *ArXiv*, abs/2402.19464, 2024. URL https://api.semanticscholar.org/CorpusID:268091304.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. *empirical methods in natural language processing*, 2014.

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, and Ling Liu. Lazy safety alignment for large language models against harmful fine-tuning. 2024a. URL https://api.semanticscholar.org/CorpusID:270095345.

Youcheng Huang, Jingkun Tang, Duanyu Feng, Zheng Zhang, Wenqiang Lei, Jiancheng Lv, and Anthony G Cohn. Dishonesty in helpful and harmless alignment. *arXiv preprint arXiv:2406.01931*, 2024b.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. Baseline defenses for adversarial attacks against aligned language models, 2023. URL https://arxiv.org/abs/2309.00614.

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

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=e2TBb5y0yFf.

Divyanshu Kumar, Anurakt Kumar, Sahil Agarwal, and Prashanth Harshangi. Increased llm vulnerabilities from fine-tuning and quantization, 2024.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K Kummerfeld, and Rada Mihalcea. A mechanistic understanding of alignment algorithms: A case study on dpo and toxicity. *arXiv preprint arXiv:2401.01967*, 2024a.

Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering, 2024b. URL https://arxiv.org/abs/2409.05907.

Chak Tou Leong, Yi Cheng, Kaishuai Xu, Jian Wang, Hanlin Wang, and Wenjie Li. No two devils alike: Unveiling distinct mechanisms of fine-tuning attacks. *ArXiv*, abs/2405.16229, 2024. URL https://api.semanticscholar.org/CorpusID:270063329.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024a.

Xuan Li, Zhanke Zhou, Jianing Zhu, Jiangchao Yao, Tongliang Liu, and Bo Han. Deepinception: Hypnotize large language model to be jailbreaker, 2024b. URL https://arxiv.org/abs/2311.03191.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7Jwpw4qKkb.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pp. 15817–15831. PMLR, 2022.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022.

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

ShengYun Peng, Pin-Yu Chen, Matthew Hull, and Duen Horng Chau. Navigating the safety landscape: Measuring risks in finetuning large language models, 2024. URL https://arxiv.org/abs/2405.17374.

Ethan Perez, Sam Ringer, Kamilė Lukošiūtė, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, Andy Jones, Anna Chen, Ben Mann, Brian Israel, Bryan Seethor, Cameron McKinnon, Christopher Olah, Da Yan, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Guro Khundadze, Jackson Kernion, James Landis, Jamie Kerr, Jared Mueller, Jeeyoon Hyun, Joshua Landau, Kamal Ndousse, Landon Goldberg, Liane Lovitt, Martin Lucas, Michael Sellitto, Miranda Zhang, Neerav Kingsland, Nelson Elhage, Nicholas Joseph, Noemí Mercado, Nova DasSarma, Oliver Rausch, Robin Larson, Sam McCandlish, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Jack Clark, Samuel R. Bowman, Amanda Askell, Roger Grosse, Danny Hernandez, Deep Ganguli, Evan Hubinger, Nicholas Schiefer, and Jared Kaplan. Discovering language model behaviors with model-written evaluations, 2022.

Mansi Phute, Alec Helbling, Matthew Daniel Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. LLM self defense: By self examination, LLMs know they are being tricked. In *The Second Tiny Papers Track at ICLR 2024*, 2024. URL https://openreview.net/forum?id=YoqgcIA19o.

Xiangyu Qi, Kaixuan Huang, Ashwinee Panda, Peter Henderson, Mengdi Wang, and Prateek Mittal. Visual adversarial examples jailbreak aligned large language models, 2023a.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *ArXiv*, abs/2310.03693, 2023b. URL https://api.semanticscholar.org/CorpusID:263671523.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

Shaina Raza, Oluwanifemi Bamgbose, Shardul Ghuge, Fatemeh Tavakoli, and Deepak John Reji. Developing safe and responsible large language models–a comprehensive framework. *arXiv preprint arXiv:2404.01399*, 2024.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition. *arXiv preprint arXiv:2312.06681*, 2023.

Domenic Rosati, Jan Wehner, Kai Williams, Lukasz Bartoszcze, Jan Batzner, Hassan Sajjad, and Frank Rudzicz. Immunization against harmful fine-tuning attacks. In *Conference on Empirical Methods in Natural Language Processing*, 2024. URL https://api.semanticscholar.org/CorpusID:268032044.

Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. *arXiv preprint arXiv:2308.01263*, 2023.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. *arXiv: Computation and Language*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Esin DURMUS, Zac Hatfield-Dodds, Scott R Johnston, Shauna M Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tvhaxkMKAn.

Zhouxing Shi, Yihan Wang, Fan Yin, Xiangning Chen, Kai-Wei Chang, and Cho-Jui Hsieh. Red teaming language model detectors with language models. *Transactions of the Association for Computational Linguistics*, 12:174–189, 2023. URL https://api.semanticscholar.org/CorpusID:258987266.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Dylan Schrader, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-López, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kocoń, Jana Thompson, Janelle Wingfield, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan Batchelder, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Guerr, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi,

Kory Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Colón, Luke Metz, Lütfi Kerem Şenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramírez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swędrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mitch Walker, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T, Nanyun Peng, Nathan A. Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nicole Martinez, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramon Risco, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima, Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsu Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, Zhengliang Liu, Yixin Liu, Yijue Wang, Zhikun Zhang, Bertie Vidgen, Bhavya Kailkhura, Caiming Xiong, Chaowei Xiao, Chunyuan Li, Eric Xing, Furong Huang, Hao Liu, Heng Ji, Hongyi Wang, Huan Zhang, Huaxiu Yao, Manolis Kellis, Marinka Zitnik, Meng Jiang, Mohit Bansal, James Zou, Jian Pei, Jian Liu, Jianfeng Gao, Jiawei Han, Jieyu Zhao, Jiliang Tang, Jindong Wang, Joaquin Vanschoren, John Mitchell, Kai Shu, Kaidi Xu, Kai-Wei Chang, Lifang He, Lifu Huang, Michael Backes, Neil Zhenqiang Gong, Philip S. Yu, Pin-Yu Chen, Quanquan Gu, Ran Xu, Rex Ying, Shuiwang Ji, Suman Jana, Tianlong Chen, Tianming Liu, Tianyi Zhou, William Wang, Xiang Li, Xiangliang Zhang, Xiao Wang, Xing Xie, Xun Chen, Xuyu Wang, Yan Liu, Yanfang Ye, Yinzhi Cao, Yong Chen, and Yue Zhao. Trustllm: Trustworthiness in large language models, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *north american chapter of the association for computational linguistics*, 2018.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Alex Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*, 2023.

Guanchu Wang, Yu-Neng Chuang, Ruixiang Tang, Shaochen Zhong, Jiayi Yuan, Hongye Jin, Zirui Liu, Vipin Chaudhary, Shuai Xu, James Caverlee, and Xia Hu. Taylor unswift: Secured weight release for large language models via Taylor expansion. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 6928–6941, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-main.393.

Jiong Wang, Jiazhao Li, Yiquan Li, Xiangyu Qi, Junjie Hu, Yixuan Li, Patrick Drew McDaniel, Muhao Chen, Bo Li, and Chaowei Xiao. Mitigating fine-tuning jailbreak attack with backdoor enhanced alignment. *ArXiv*, abs/2402.14968, 2024b. URL https://api.semanticscholar.org/CorpusID:267897454.

Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. Detoxifying large language models via knowledge editing, 2024c.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, et al. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*, 2023a.

Pengyu Wang, Dong Zhang, Linyang Li, Chenkun Tan, Xinghao Wang, Ke Ren, Botian Jiang, and Xipeng Qiu. Inferaligner: Inference-time alignment for harmlessness through cross-model guidance. *arXiv preprint arXiv:2401.11206*, 2024d.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions, 2023b. URL https://arxiv.org/abs/2212.10560.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *arXiv preprint arXiv:2307.02483*, 2023.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022a.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Zeming Wei, Yifei Wang, Ang Li, Yichuan Mo, and Yisen Wang. Jailbreak and guard aligned language models with only few in-context demonstrations, 2024. URL https://arxiv.org/abs/2310.06387.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2550–2575, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.167. URL https://aclanthology.org/2023.findings-emnlp.167.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Kang Liu, and Jun Zhao. Mastering symbolic operations: Augmenting language models with compiled neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=9nsNyN0vox.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. Reft: Representation finetuning for language models. *arXiv preprint arXiv:2404.03592*, 2024a.

Zhengxuan Wu, Atticus Geiger, Aryaman Arora, Jing Huang, Zheng Wang, Noah D. Goodman, Christopher D. Manning, and Christopher Potts. pyvene: A library for understanding and improving PyTorch models via interventions. 2024b. URL arxiv.org/abs/2403.07809.

Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao Wu. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, 5:1486–1496, 2023. URL https://api.semanticscholar.org/CorpusID:266289038.

Zhihao Xu, Ruixuan Huang, Xiting Wang, Fangzhao Wu, Jing Yao, and Xing Xie. Uncovering safety risks in open-source llms through concept activation vector. *arXiv preprint arXiv:2404.12038*, 2024a.

Zihao Xu, Yi Liu, Gelei Deng, Yuekang Li, and Stjepan Picek. Llm jailbreak attack versus defense techniques–a comprehensive study. *arXiv preprint arXiv:2402.13457*, 2024b.

J. Yang et al. Red teaming language models via activation engineering. *LessWrong*, 2023a. URL https://www.lesswrong.com/posts/iHmsJdxgMEWmAfNne/red-teaming-language-models-via-activation-engineering.

Xianjun Yang, Xiao Wang, Qi Zhang, Linda Ruth Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. Shadow alignment: The ease of subverting safely-aligned language models. *ArXiv*, abs/2310.02949, 2023b. URL https://api.semanticscholar.org/CorpusID:263620436.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, 4(2):100211, June 2024. ISSN 2667-2952. doi: 10.1016/j.hcc.2024.100211. URL http://dx.doi.org/10.1016/j.hcc.2024.100211.

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

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Jiahao Xu, Tian Liang, Pinjia He, and Zhaopeng Tu. Refuse whenever you feel unsafe: Improving safety in llms via decoupled refusal training. *arXiv preprint arXiv:2407.09121*, 2024.

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori Hashimoto, and Daniel Kang. Removing rlhf protections in gpt-4 via fine-tuning. *ArXiv*, abs/2311.05553, 2023. URL https://api.semanticscholar.org/CorpusID:265067269.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. A comprehensive study of knowledge editing for large language models, 2024a. URL https://arxiv.org/abs/2401.01286.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey, 2024b. URL https://arxiv.org/abs/2308.10792.

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

Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. On prompt-driven safeguarding for large language models. 2024. URL https://api.semanticscholar.org/CorpusID:267334949.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.

Minjun Zhu, Linyi Yang, and Yue Zhang. Personality alignment of large language models. *arXiv preprint arXiv:2408.11779*, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023b. URL https://arxiv.org/abs/2307.15043.

Andy Zou, Long Phan, Justin Wang, Derek Duenas, Maxwell Lin, Maksym Andriushchenko, J Zico Kolter, Matt Fredrikson, and Dan Hendrycks. Improving alignment and robustness with circuit breakers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=IbIB8SBKFV.

A Appendix

A.1 Limitations

While SafetyLock demonstrates promising results in maintaining the safety of fine-tuned language models, it is important to acknowledge several limitations. Primarily, SafetyLock requires access to both model weights and intermediate activations for implementation, which may limit its applicability in scenarios where such access is restricted or unavailable. Additionally, the method employs a symmetric locking mechanism; consequently, if an unauthorized party gains access to the model weights or activation values, they could potentially reverse-engineer the process to unlock and bypass SafetyLock's protections. Lastly, while SafetyLock shows strong performance against current attack methods, its long-term robustness against evolving adversarial techniques remains to be studied. These limitations present opportunities for future work to enhance and expand the capabilities of SafetyLock, ensuring its continued effectiveness in maintaining AI safety.

A.2 Consistency of Harmlessness Directions in Fine-tuned Models

To validate SafetyLock's effectiveness, we conducted a comprehensive analysis of the original Llama-3-Instruct 8B model and its fine-tuned versions under various risk levels. Our experimental setup was as follows:

We first extracted activation values from the 31st layer, 26th head of the Llama-3-8B Instruct model, which we identified as the most sensitive to harmlessness through linear regression, achieving the

highest binary classification accuracy. We then performed forward computation on a safety dataset, saving the activation values of the last token for both safe and unsafe samples. Using 2D PCA for dimensionality reduction, we visualized the shift in activation values between safe and unsafe samples by connecting their center points with arrows, illustrating both the direction and magnitude of the shift.

Remarkably, we observed high similarity in these shifts across different risk levels (i.e., fine-tuning on data from different domains). To quantitatively assess the similarity between the safety directions found in the original model and those in the fine-tuned models, we employed KL divergence:

$$D_{KL}(P||Q) = \sum_i P(i) \log \left(\frac{P(i)}{Q(i)} \right) \tag{4}$$

where P and Q represent the distributions of safety directions in the original and fine-tuned models, respectively.

To further illustrate the change in similarity during the fine-tuning process, we employed one-dimensional linear interpolation of weights (Peng et al., 2024). This method allows us to smoothly transition from the original model weights to the fine-tuned model weights, providing insight into how the safety directions evolve during the fine-tuning process. The interpolation is defined as:

$$\theta_\alpha = \theta + \alpha(\theta' - \theta) \tag{5}$$

where θ represents the weights of the original Llama-3 model, θ' the weights of the fine-tuned model, and $\alpha \in [-0.2, 1.2]$ is the interpolation parameter. We extend α slightly beyond the [0, 1] range to observe trends slightly before and after the actual interpolation points.

The interpolation process is implemented as follows:

1. We first extract the state dictionaries of both the base model (θ) and the fine-tuned model (θ').
2. For each layer, we compute the difference vector: $d_1 = \theta' - \theta$.
3. We then create new weights for each α value: $\theta_\alpha = \theta + \alpha d_1$.
4. These new weights are used to reconstruct a new state dictionary, maintaining the original structure and naming conventions of the model.

We use these interpolated models to compute the KL divergence between the safety directions of the original model and the interpolated models at each step. This results in a smooth curve showing how the similarity of safety directions changes as the model transitions from its original state to the fine-tuned state.

B Mathematical Explanation of SafetyLock's Effectiveness in Suppressing Harmful Outputs

In this section, we provide a mathematical justification for why SafetyLock can extract transferable safety directions from the original language model and effectively apply them to fine-tuned models to suppress harmful outputs. Our explanation is grounded in the properties of Transformer-based language models and the nature of fine-tuning on limited datasets.

B.1 Activation Space and Safety Directions

Let us denote the activations of the original (pre-fine-tuned) language model at layer l and head h as $\mathbf{x}_{l,h} \in \mathbb{R}^D$, where D is the dimensionality of the head's output. During inference, these activations encode information about the generated tokens.

We define two sets of activations corresponding to safe and unsafe responses:

$$\mathcal{X}_{\text{safe}} = \left\{ \mathbf{x}_{l,h}^{\text{safe},i} \right\}_{i=1}^{N_{\text{safe}}}, \tag{6}$$

$$\mathcal{X}_{\text{unsafe}} = \left\{ \mathbf{x}_{l,h}^{\text{unsafe},i} \right\}_{i=1}^{N_{\text{unsafe}}}, \tag{7}$$

where N_{safe} and N_{unsafe} are the numbers of safe and unsafe samples, respectively.

We compute the *safety direction* $\boldsymbol{\theta}_{l,h} \in \mathbb{R}^D$ as the mean difference between the activations for safe and unsafe responses:

$$\boldsymbol{\theta}_{l,h} = \frac{1}{N_{\text{safe}}} \sum_{i=1}^{N_{\text{safe}}} \mathbf{x}_{l,h}^{\text{safe},i} - \frac{1}{N_{\text{unsafe}}} \sum_{i=1}^{N_{\text{unsafe}}} \mathbf{x}_{l,h}^{\text{unsafe},i}. \tag{8}$$

This vector represents the average shift in activation space needed to move from an unsafe response towards a safe one.

B.2 Preservation of Safety Directions During Fine-Tuning

Fine-tuning a language model on a new dataset modifies its parameters to adapt to specific tasks or domains. However, when the fine-tuning dataset is limited in size or scope, the changes to the model's internal representations are often localized and do not significantly alter the global structure of the activation space (Golechha & Dao, 2024; Godfrey et al., 2022).

Let $\tilde{\mathbf{x}}_{l,h}$ denote the activations of the fine-tuned model at layer l and head h. Empirically, we observe that there exists a strong linear relationship between the activations of the original and fine-tuned models:

$$\tilde{\mathbf{x}}_{l,h} \approx \mathbf{x}_{l,h} + \Delta \mathbf{x}_{l,h}, \tag{9}$$

where $\Delta \mathbf{x}_{l,h}$ represents the change in activations due to fine-tuning, which is relatively small in magnitude compared to $\mathbf{x}_{l,h}$ for many dimensions.

Moreover, the safety direction $\boldsymbol{\theta}_{l,h}$ computed from the original model remains relevant in the fine-tuned model because the relative differences between safe and unsafe activations are preserved:

$$\tilde{\boldsymbol{\theta}}_{l,h} = \left(\tilde{\mathbf{x}}_{l,h}^{\text{safe}} - \tilde{\mathbf{x}}_{l,h}^{\text{unsafe}} \right) \approx \left(\mathbf{x}_{l,h}^{\text{safe}} - \mathbf{x}_{l,h}^{\text{unsafe}} \right) = \boldsymbol{\theta}_{l,h}. \tag{10}$$

This approximation holds under the assumption that fine-tuning does not disproportionately affect the dimensions critical for encoding safety-related information.

B.3 Effectiveness of Activation Intervention

During inference with the fine-tuned model, we intervene by adjusting the activations along the safety direction:

$$\tilde{\mathbf{x}}_{l,h}^{\text{intervened}} = \tilde{\mathbf{x}}_{l,h} + \alpha \left(\boldsymbol{\sigma}_{l,h} \odot \boldsymbol{\theta}_{l,h} \right), \tag{11}$$

where:

- $\alpha \in \mathbb{R}$ is the scaling factor controlling the intensity of the intervention.
- $\boldsymbol{\sigma}_{l,h} \in \mathbb{R}^D$ is the standard deviation vector of activations along each dimension, capturing the typical variability.
- $\odot$ denotes element-wise multiplication.

This adjustment effectively shifts the activations towards regions in the activation space associated with safe responses. Since the safety direction $\boldsymbol{\theta}_{l,h}$ is approximately preserved in the fine-tuned model, this intervention remains effective.

B.4 Impact on Output Probabilities

The language model generates the next token based on a probability distribution computed from the final activations. Adjusting the activations as in Equation equation 11 influences the logits $\mathbf{z} \in \mathbb{R}^V$ (where V is the vocabulary size) before the softmax function:

$$\mathbf{z}^{\text{intervened}} = \mathbf{z} + W_{\text{head}} \left(\alpha \left(\boldsymbol{\sigma}_{l,h} \odot \boldsymbol{\theta}_{l,h} \right) \right), \tag{12}$$

where $W_{\text{head}} \in \mathbb{R}^{V \times D}$ is the weight matrix projecting activations to logits.

The adjustment $\Delta \mathbf{z} = W_{\text{head}} \left(\alpha \left(\boldsymbol{\sigma}_{l,h} \odot \boldsymbol{\theta}_{l,h} \right) \right)$ biases the logits towards tokens that are more likely in safe responses and away from those prevalent in unsafe responses.

B.5 Suppressing Harmful Outputs

The probability of generating a harmful token t_{harm} is given by:

$$P(t_{\text{harm}}) = \frac{\exp\left(z_{t_{\text{harm}}}^{\text{intervened}}\right)}{\sum_{i=1}^{V} \exp\left(z_i^{\text{intervened}}\right)}. \tag{13}$$

By decreasing $z_{t_{\text{harm}}}^{\text{intervened}}$ relative to other logits, we reduce $P(t_{\text{harm}})$. Since the intervention shifts the activations towards safe regions, the logits for harmful tokens are decreased, and the model is less likely to generate harmful outputs.

B.6 Transferability Across Models

The key to SafetyLock's transferability lies in the similarity of safety directions between the original and fine-tuned models. Since the fine-tuning process does not significantly alter the relative positions of safe and unsafe activations in the activation space (as per Equation equation 10), the safety directions computed from the original model remain effective when applied to the fine-tuned model.

This property is supported by empirical observations of low Kullback–Leibler (KL) divergence between the activation distributions of the original and fine-tuned models (see Figure 2 in Section 3.3). The minimal divergence indicates that the overall structure of the activation space, especially along dimensions relevant to safety, is preserved during fine-tuning.

B.7 Conclusion

Mathematically, SafetyLock leverages the preserved safety directions in the activation space to adjust the model's internal computations towards generating safe outputs. By intervening along these directions, we effectively suppress harmful responses without requiring retraining or fine-tuning of the model. The minimal changes to the activation distributions during fine-tuning ensure that the safety directions remain applicable, allowing for efficient and transferable safety interventions across different models and fine-tuning scenarios.

This theoretical explanation provides a foundation for understanding the effectiveness of SafetyLock in suppressing harmful outputs while maintaining the model's overall performance on benign tasks.

C The Risks of Fine-tuning LLMs and Experimental Setup

HEx-PHI (Qi et al., 2023b) is based on 11 categories of prohibited use cases merged from Meta's Llama-3 acceptable use policy and OpenAI's usage policies: (1) Illegal Activity, (2) Child Abuse Content, (3) Hate, Harass, Violence, (4) Malware, (5) Physical Harm, (6) Economic Harm, (7) Fraud, Deception, (8) Adult Content, (9) Political Campaigning, (10) Privacy Violation Activity, and (11) Tailored Financial Advice. The dataset includes 30 examples per category, totaling 330 examples. This ensures a comprehensive safety evaluation aligned with industry-standard usage policies.

For Risk-1, we use negative samples from the HH-RLHF preference dataset. We select 10, 100, 1000, and 10000 samples respectively and trained for 5 epochs with a learning rate of 2×10^{-5}. For Risk-2,

we use 10 samples from Qi et al. (2023b) and trained for 5 epochs with a learning rate of 2×10^{-5}. For Risk-3, we use the first 50,000 samples from the Alpaca dataset (Wang et al., 2023b) and trained for 5 epochs with a learning rate of 2×10^{-5} [1]. We set the last token $r = 5$.

Recognizing the potential of existing approaches to address safety issues in fine-tuned language models, we conducted comparative analyses across two categories as the same time: training-based and inference-time methods. For training-based approaches, we evaluated PPO, DPO, SFT (with safety data mixed during fine-tuning), SFT (with safety data mixed post-fine-tuning), and model-editing. Inference-time methods included ICD, PPL, Paraphrase, Retokenization, Safe-Reminder, and Self-Exam. These methods were assess based on efficiency, attack sample rejection rate, and normal text rejection rate, providing a comprehensive evaluation of their effectiveness in maintaining model safety while preserving functionality. This multi-faceted approach allows us to rigorously examine the trade-offs between safety and performance.

Specifically, to ensure reproducibility, we followed past experimental settings and use 2000 safety data points from Bianchi et al. (2024) for SFT experiments. We considered two experimental settings for SFT. The first is After Training, which simulates the scenario where safety disappears after fine-tuning the language model and needs to be restored. This applies to all fine-tuned language models. The second is During Training, which simulates starting from the original model and requiring the mixing of additional safety data during training to prevent safety disappearance. However, the limitation of this method is that it still requires retraining for already fine-tuned language models. For PPO, we also use 2000 samples from Bianchi et al. (2024), and we use LlamaGuard-7b (Bhatt et al., 2023) as the Reward model. For DPO, based on the 2000 samples, we use samples generated by the fine-tuned language model (almost all of which are harmful) as negative samples for training. For the Model-Edited method, we use the most common Detoxifying with Intraoperative Neural Monitoring (DINM) method and followed the original setup using SafeEdit data[2] for editing.

D Additional Experiments

D.1 Analysis of SafetyLock's Intervention

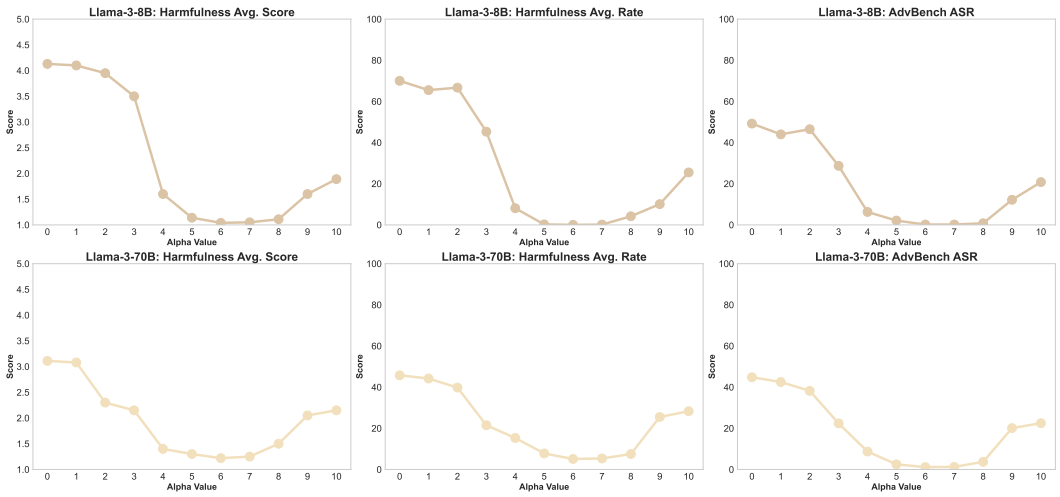

Figure 7: Impact of SafetyLock's intervention distance (α) on model safety metrics for Llama-3-8B and Llama-3-70B models. The graphs show Harmfulness Average Score, Harmfulness Average Rate, and AdvBench ASR across different α values. Note that for these experiments, the intervention degree K is set to 24, indicating the number of attention heads influenced by SafetyLock.

Distance α. Our experimental results, as illustrated in Figure 7, demonstrate the significant influence of SafetyLock's intervention distance (α) on model safety across different model sizes. For both

[1] We use the official fine-tuning code https://github.com/meta-llama/llama-recipes

[2] https://huggingface.co/datasets/zjunlp/SafeEdit

Llama-3-8B and Llama-3-70B, we observe a clear U-shaped trend in harmfulness metrics as α increases. Initially, as α rises from 0 to 4, there's a sharp decrease in harmfulness scores and rates, as well as the AdvBench ASR. This indicates that moderate intervention effectively enhances model safety. However, beyond $\alpha = 4$, we see a gradual increase in these metrics, suggesting that excessive intervention may lead to unintended consequences, potentially disrupting the model's learned safety boundaries. Notably, Llama-3-70B exhibits more stability across different α values compared to Llama-3-8B, implying that larger models may be more resilient to intervention adjustments. These findings underscore the importance of carefully calibrating SafetyLock's intervention parameters to achieve optimal safety improvements while maintaining model performance, with an optimal α value around 4-6 for both model sizes.

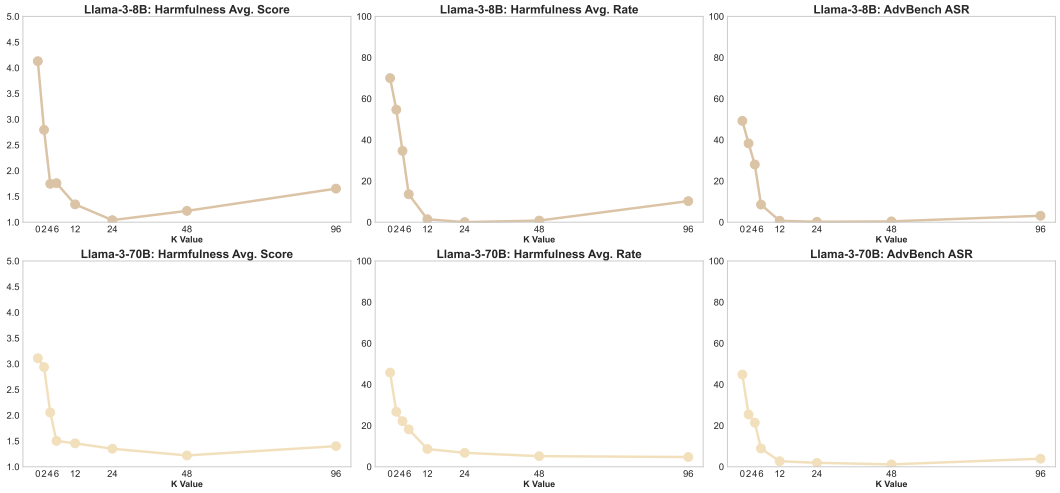

Figure 8: Impact of SafetyLock's intervention degree (K) on model safety metrics for Llama-3-8B and Llama-3-70B models. The graphs illustrate the Harmfulness Average Score, Harmfulness Average Rate, and AdvBench ASR across different K values, ranging from 0 to 96. Lower scores indicate better safety performance. Note the rapid improvement in safety metrics as K increases from 0 to 6, followed by more gradual enhancements up to K=24, with a slight uptick at K=96 for some metrics.

Degree K. Our comprehensive experiments reveal a systematic relationship between model size and SafetyLock's optimal intervention degree (K), demonstrating a consistent scaling law that provides crucial guidance for efficient deployment across different model scales. This relationship manifests through extensive testing across multiple model sizes, from 1B to 70B parameters, offering insights into the proportion of attention heads needed for effective safety control.

Table 3: Impact of K on 1B-scale Model Safety

K Value	AdvBench ASR
Vanilla	21.15%
K=3	16.54%
K=6	**10.65%**
K=12	11.08%
K=24	12.44%
K=48	47.50%

Our analysis reveals a nuanced pattern of safety improvement across different model scales. For Llama-3-8B and Llama-3-70B, we observe a rapid enhancement in safety metrics as K increases from 0 to 6, followed by more gradual improvements up to K=24. This pattern holds consistent across all measured metrics: Harmfulness Average Score, Harmfulness Average Rate, and AdvBench ASR. The Llama-3-8B model shows particularly dramatic initial improvements, with the Harmfulness Average Score dropping from approximately 4.0 to 1.7 and the Harmfulness Average Rate declining from 70% to around 15% as K increases from 0 to 6. The Llama-3-70B model demonstrates similar trends

but with generally lower baseline harmfulness scores, suggesting that larger models might possess inherently stronger safety characteristics. Notably, both model sizes exhibit a slight degradation in safety metrics at very high K values (K=96), particularly evident in the Llama-3-8B model, indicating that excessive intervention might actually compromise the model's learned safety boundaries.

Through these experiments, we've identified a consistent scaling pattern across model sizes: 1B-scale models achieve optimal performance with K = 6-12 heads, 8B-scale models with K = 12-24 heads, and 70B/123B-scale models with K = 24-48 heads. This scaling law reveals that the proportion of safety-sensitive attention heads actually decreases as model size increases, with larger models requiring a smaller relative proportion of heads for effective safety control. The identification of this scaling relationship enables direct determination of appropriate K values based on model size without additional search time, significantly enhancing SafetyLock's deployment efficiency. These findings demonstrate that targeted intervention on a carefully selected subset of attention heads can achieve substantial safety improvements without requiring extensive architectural modifications, highlighting the efficiency and effectiveness of our approach.

D.2 Impact of Learning Rate on Safety Degradation

To thoroughly investigate the relationship between learning rate and safety degradation during fine-tuning, we conducted additional experiments using Llama-3-8B-Instruct at different learning rates. Following the hyperparameter settings from previous work (Qi et al., 2023b) (detailed in Appendix G.1), we initially used a learning rate of 2e-5 for our main experiments. However, considering that smaller learning rates (e.g., 1e-6) are commonly used in continued pre-training scenarios to minimize impact on model behaviors, we performed comparative experiments under Risk Level-3 fine-tuning scenario.

Table 4: Impact of Learning Rate on Safety Degradation and Recovery

Learning Rate	Vanilla ASR (%)	SafetyLock ASR (%)
2e-5	42.88	0.19
1e-6	26.92	0.00

Results in Table 4 demonstrate that a lower learning rate (1e-6) leads to less safety degradation compared to 2e-5 (26.92% vs. 42.88% ASR). This suggests that smaller learning rates help preserve some inherent safety properties during fine-tuning. Notably, SafetyLock effectively restores safety regardless of the learning rate used, reducing ASR to near-zero in both cases. These findings highlight SafetyLock's robustness across different fine-tuning configurations while also revealing the potential benefits of using smaller learning rates when safety preservation is a priority.

D.3 Direction Consistency Across Multiple Attention Heads

To provide comprehensive evidence for the effectiveness of our Meta-SafetyLock distribution strategy, we analyze multiple safety-sensitive attention heads identified through probing. Figure 9 visualizes the activation patterns in 6 representative heads - (12, 21), (14, 11), (16, 7), (16, 29), (24, 14), and (31, 26) - across the original Llama-3-8B-Instruct model and its fine-tuned variants under Risk Level-1 and Risk Level-2. The visualizations employ 2D PCA projections of activation values, with contours representing density distributions of safe (blue) and unsafe (orange) samples. Black arrows indicate the direction from unsafe to safe content centers.

Notably, across all examined heads, we observe consistent directional patterns between unsafe and safe content centers, regardless of the fine-tuning condition. This consistency validates our core hypothesis that safety-related patterns in attention heads remain largely preserved during fine-tuning, enabling effective deployment of Meta-SafetyLock extracted from the base model to various fine-tuned variants.

D.4 Domain-Specific Performance: A Case Study on Mathematical Reasoning

To rigorously evaluate SafetyLock's ability to maintain domain-specific capabilities while ensuring safety, we conducted extensive experiments using the GSM8K dataset, a challenging mathematical

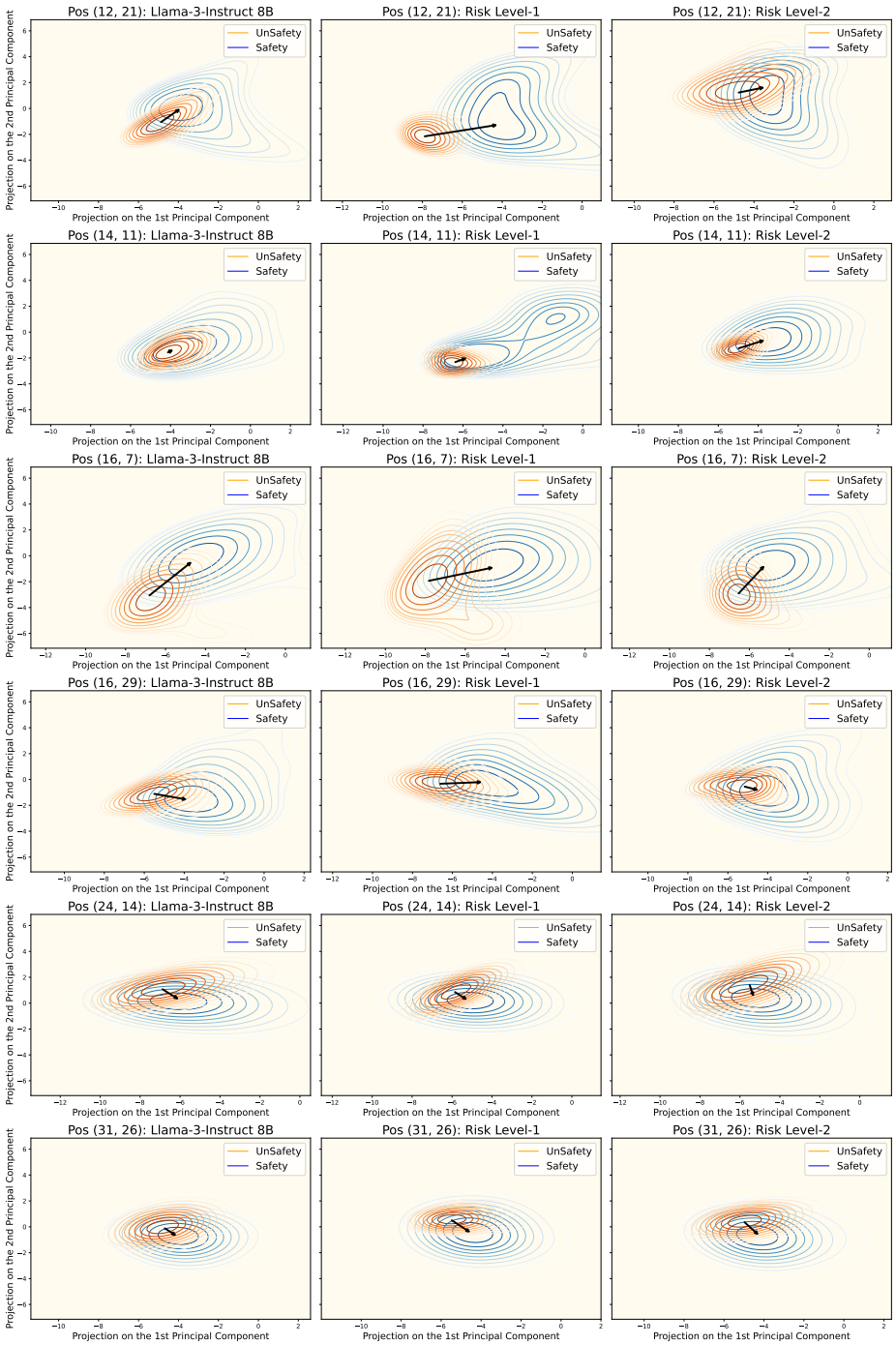

Figure 9: Visualization of activation patterns for multiple attention heads. Each row represents a different attention head position, showing consistent directional patterns across the original model and fine-tuned variants. The black arrows indicate the direction from unsafe to safe content centers, demonstrating remarkable consistency in safety directions despite fine-tuning modifications.

reasoning benchmark. We fine-tuned Llama-3-8B-Instruct on GSM8K's training set and evaluated both safety metrics and mathematical performance.

As shown in Table 5, SafetyLock demonstrates remarkable effectiveness in preserving mathematical reasoning capabilities while enhancing safety measures. The minimal performance drop in GSM8K

Table 5: Safety and Performance Metrics for GSM8K Fine-tuning

Model	AdvBench ASR	HEx-PHI Score	GSM8K Test Acc
Original	7.23%	1.45	85.59%
Model-Edited (DINM)	3.02%	1.33	5.00%
SafetyLock	**0.19%**	**1.08**	**84.91%**

accuracy (from 85.59% to 84.91%) stands in stark contrast to traditional safety-alignment methods like Model-Edited (DINM), which suffers catastrophic degradation to 5.00% accuracy. Simultaneously, SafetyLock achieves superior safety metrics, reducing AdvBench ASR from 7.23% to 0.19% and improving the HEx-PHI Score from 1.45 to 1.12. These results provide compelling evidence that SafetyLock can successfully maintain domain-specific capabilities while ensuring robust safety guardrails, addressing a critical challenge in deploying safe and effective language models for specialized tasks.

D.5 Impact of Activation Normalization on SafetyLock

To investigate the role of activation normalization in SafetyLock, we conducted experiments comparing the performance with and without the standard deviation term σ_l^h in Equation 4. When omitting σ_l^h, we set it to 1, effectively removing the activation-specific scaling of interventions.

Table 6: Impact of Activation Normalization on Safety and Performance

Model	AdvBench ASR	HEx-PHI Score	GSM8K Test Acc
Original	7.23%	1.45	85.59%
SafetyLock w/o σ_l^h	**0.0%**	**1.03**	52.24%
SafetyLock w/ σ_l^h	0.19%	1.12	**84.91%**

Results in Table 6 demonstrate the critical role of σ_l^h in balancing safety and model utility. Without normalization, while safety metrics improve marginally (ASR: 0.0%, HEx-PHI: 1.03), the model suffers severe performance degradation on GSM8K (52.24%). Including σ_l^h maintains strong safety improvements while preserving the model's mathematical reasoning capabilities (84.91% accuracy). This suggests that activation-specific scaling through σ_l^h is essential for preventing over-aggressive interventions that could compromise model functionality. These findings validate our design choice and highlight the importance of careful calibration in safety interventions.

D.6 Comparison with Circuit Breakers

We compare SafetyLock with Circuit Breakers (Zou et al., 2024), a recent approach from NeurIPS 2024 that builds upon Representation Engineering techniques (Zou et al., 2023a) to remap harmful representations towards incoherent or refusal states. Using three fine-tuned versions of Llama-3-8B-Instruct with consistent hyperparameters, we observe significant performance differences across risk levels.

Table 7 presents results for the three risk scenarios. For Level-1 (explicitly harmful fine-tuning), SafetyLock reduces AdvBench ASR to 0.19% and HEx-PHI Score to 1.36, while Circuit Breakers shows increased vulnerability (ASR: 84.62%, Score: 3.62). In Level-2 scenarios (implicitly harmful fine-tuning), both methods demonstrate improvement over the baseline, though SafetyLock achieves superior results (ASR: 5.19% vs 27.12%). For Level-3 (benign fine-tuning), Circuit Breakers exhibits significant degradation (ASR: 94.04%) while SafetyLock maintains robust performance (ASR: 0.19%).

For comprehensive evaluation, we also assess both methods on Circuit Breakers' original benchmark scenarios, as shown in Table 8. SafetyLock achieves perfect defense (0% ASR) across all attack types, surpassing Circuit Breakers' performance on its own evaluation metrics.

Table 7: Comparison with Circuit Breakers across different risk levels using Llama-3-8B-Instruct

Method	Level-1 (ASR/Score)	Level-2 (ASR/Score)	Level-3 (ASR/Score)
Original Fine-tuned	49.24%/4.13	38.46%/3.19	42.88%/3.23
Circuit Breakers	84.62%/3.62	27.12%/2.10	94.04%/3.79
SafetyLock	**0.19%/1.36**	**5.19%/1.07**	**0.19%/1.04**

Table 8: Performance on Circuit Breakers' original benchmark scenarios

Method	AutoDAN	PAIR	GCG
Base Model	3.7%	18.7%	44.5%
Circuit Breakers	0.0%	7.5%	2.5%
SafetyLock	**0.0%**	**0.0%**	**0.0%**

Regarding computational efficiency, SafetyLock requires 5 minutes for Meta-SafetyLock construction and 0.1 seconds for distribution to each fine-tuned model. In contrast, Circuit Breakers demands 22 minutes 15 seconds per model on an A100. This significant efficiency advantage, combined with superior safety metrics, demonstrates SafetyLock's practical advantages for large-scale deployment scenarios.

The performance disparity may be attributed to Circuit Breakers' representation remapping strategy being less effective when model safety boundaries have been substantially modified through fine-tuning. SafetyLock's approach of targeting specific attention heads appears more robust to such modifications while maintaining computational efficiency.

D.7 Impact of Token Window Size on SafetyLock

The choice of how many final tokens to consider when calculating safety directions represents a crucial design decision in SafetyLock's implementation. While previous works often use the entire hidden state for intervention, we hypothesized that focusing on a smaller window of final tokens might capture safety-relevant patterns more effectively while maintaining computational efficiency.

Table 9: Impact of Token Window Size (r) on Safety Performance

Model	AdvBench ASR (%)		
	Level-1	Level-2	Level-3
Vanilla	49.24	38.46	42.88
r = 1	1.14	6.84	3.61
r = 3	0.76	8.55	0.19
r = 5	**0.19**	**5.19**	**0.19**
r = 10	0.48	8.08	0.57

To determine the optimal token window size, we conducted extensive experiments varying r from 1 to 10 tokens across all three risk levels, as shown in Table 9. Our findings reveal that r = 5 consistently achieves optimal or near-optimal safety performance across all scenarios. While smaller windows (r = 1, 3) can effectively improve safety, they may not capture sufficient context for robust intervention. Conversely, larger windows (r = 10) show slightly degraded performance, possibly due to including less relevant contextual information. This empirical evidence supports our choice of r = 5 as the default parameter, offering the best balance between robust safety improvement and effective intervention across different fine-tuning scenarios.

E RECOMMENDATIONS FOR DEPLOYING SAFETYLOCK

Understanding the diverse landscape of model deployment scenarios is crucial for effectively implementing SafetyLock to maintain safety while enabling customization. The method's effectiveness and implementation strategy vary significantly depending on the model's distribution approach and user priorities, leading to distinct considerations for different deployment contexts.

For closed-source models served through APIs (e.g., GPT-4), SafetyLock offers an optimal solution through seamless integration into the service provider's infrastructure. Model providers can automatically apply SafetyLock after each fine-tuning operation, ensuring consistent safety standards while maintaining customization capabilities. This approach particularly benefits enterprises in regulated industries that require both task-specific optimization and strict safety controls, as it preserves the ability to customize models for specific use cases without compromising safety standards. The automated application of SafetyLock in this context ensures that all model variants maintain robust safety guardrails, regardless of the extent of customization.

In scenarios involving open-source models with safety-conscious users, SafetyLock can be effectively implemented as part of the standard deployment pipeline. Organizations using open-source models can apply SafetyLock during their model serving phase, maintaining safety controls while preserving the benefits of customization. This implementation strategy allows organizations to balance the flexibility inherent in open-source models with the need for robust safety guarantees, ensuring that fine-tuned models remain both useful and safe. Safety-conscious users can leverage SafetyLock to maintain consistent safety standards across their deployments while still benefiting from the customization capabilities that open-source models provide.

To address the fundamental challenge of malicious users with full access to open-source weights, we propose a hybrid deployment strategy that combines transparency with controlled access to safety-critical components. This approach involves open-sourcing the majority of model weights while retaining control of a small subset of safety-critical weights using methods like Taylor Unswift (Wang et al., 2024a). By providing efficient access to these controlled weights through a service API and applying SafetyLock during the serving phase, organizations can maintain crucial safety controls while preserving the benefits of open-source accessibility. This balanced solution ensures that users can customize models for their specific needs without easily circumventing safety measures, as the critical safety-related parameters remain protected under controlled access.

For successful implementation, organizations should establish comprehensive monitoring systems to regularly update safety vectors, implement automatic safety checks post-fine-tuning, and develop clear protocols for handling potential conflicts between safety measures and legitimate use cases. Regular assessment and updating of safety mechanisms ensure that SafetyLock remains effective against evolving harmful behaviors, while clear documentation and guidelines help users understand the implications and importance of these safety measures. Through these carefully considered deployment strategies and best practices, SafetyLock provides a robust framework for maintaining model safety across various deployment scenarios, acknowledging and addressing the inherent challenges in protecting open-source models while enabling their beneficial applications.