TRANSFERABLE LLM MISALIGNMENT ATTACK VIA SAFETY TASK VECTOR EXTRACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Safety alignment is critical to ensure that large language models appropriately respond to harmful instructions while remaining helpful on benign queries. Existing works on misalignment attacks have demonstrated that safety alignment can be easily compromised for open-weight models. However, these attacks are typically constructed for a specific target model and need to be reconstructed when the target changes. In this paper, we introduce a task arithmetic-based attack that extracts a transferable safety parameter direction that can be reused across multiple post-trained models derived from the same pretrained base model. To construct this direction, we first train paired models with and without safety-related data and extract the safety-related task vector through task arithmetic. Then, we utilize multi-seed sign consensus to suppress training noise and retain stable safety-related updates. The resulting safety task vector can be subtracted from various models derived from the same base to weaken their safety behavior. Extensive experiments across multiple model families demonstrate that our method effectively compromises safety alignment while largely preserving general model utility and exhibits transferability among models sharing the same pretrained base.

1 INTRODUCTION

Safety alignment of large language models (LLMs) has become a critical requirement as they are increasingly integrated into real-world systems and interact directly with users. A safety-aligned LLM should learn to remain helpful on benign queries while appropriately refusing or constraining its responses to harmful instructions. To develop such behavior, modern LLM development pipelines usually incorporate substantial safety training data across multiple post-training procedures, including supervised fine-tuning (Wei et al., 2021), preference optimization (Rafailov et al., 2023), and reinforcement learning (Ouyang et al., 2022).

However, safety alignment can be easily compromised for open-weight models. Recent studies have demonstrated this vulnerability through various forms of misalignment attacks. Fine-tuning-based attacks show that aligned models can lose their safeguards after being trained on a small number of malicious examples (Yang et al., 2023; Betley et al., 2025; Russinovich et al., 2026). Activation steering-based attacks identify and remove activation directions associated with safety refusals (Arditi et al., 2024; Korznikov et al., 2025; Weidmann, 2025). Despite their effectiveness, these attacks are typically constructed for a single target model and need to be reconstructed when the target changes. In practice, however, many open-weight models are post-trained variants of the same pretrained base model. This raises a broader question: can an attack constructed once be reused across multiple post-trained models sharing the same pretrained base?

Motivated by this observation, we investigate whether the safety behavior of LLMs can be captured by a transferable direction in parameter space. To this end, we explore task arithmetic (Ilharco et al., 2023) as a mechanism for constructing transferable parameter-space directions. In task arithmetic, a task vector represents the parameter change induced by fine-tuning a pretrained model on a particular task or dataset. Prior works have shown that such vectors can be added, subtracted, or rescaled to control task-specific behaviors (Ilharco et al., 2023; Yadav et al., 2023; Ortiz-Jimenez et al., 2023). This formulation is particularly suitable for models derived from the same pretrained base, as their shared base model provides a common reference space to apply the same task vector. Consequently, unlike existing attacks that need to be reconstructed for each target model, task

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

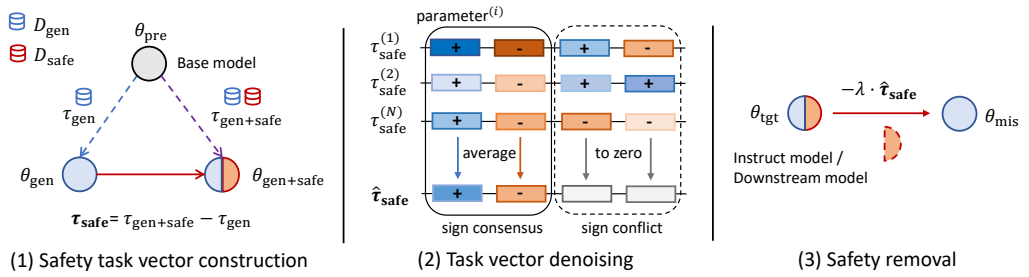

Figure 1: A depiction of the steps involved in our attack. (1) Safety task vector construction. We first construct paired models trained with and without safety-related data, and obtain a safety task vector by subtracting their task vectors. (2) Task vector denoising. We extract N safety task vectors across multiple training seeds and retain only parameters whose update signs are consistent across all runs. (3) Safety removal. We finally subtract the denoised safety task vector from the target model to remove its safety alignment.

arithmetic provides the possibility of extracting a safety-related direction once and transferring it across multiple post-trained variants. This setting is especially relevant to open-weight model families such as Llama (Grattafiori et al., 2024) and Qwen (Yang et al., 2025), where publicly available pretrained base models provide the reference for task vector extraction, while diverse post-trained variants constitute potential targets for its application.

In this paper, we introduce a task arithmetic-based framework for extracting transferable safety task vectors and constructing misaligned models. Specifically, we construct paired models trained with and without safety-related data, and obtain a safety task vector by task arithmetic. To further suppress the training noise, we repeat this paired training process across multiple seeds and retain only parameters whose update signs are consistent across all runs. The resulting vector can then be subtracted from a target aligned model to weaken its safety behavior. This design provides three key advantages. (1) Task vectors naturally support parameter-space transfer, allowing the extracted safety vector to be applied to all models derived from the same base model without additional training. (2) Paired differencing suppresses general-task-related components, while sign consensus filters the training noise. Together, they produce a more safety-related and stable task vector with less interference with model utility. (3) The entire procedure requires no harmful training responses, as the misalignment direction is obtained only by reversing the extracted safety direction.

We visualize the steps involved in our attack in Figure 1. Our contributions are as follows:

- We study the transferability of safety direction in LLM parameter space, showing that models post-trained from a shared pretrained base can be vulnerable to a safety direction extracted only once.
- We propose a safety task vector extraction method based on paired task vector differencing and adopt the multi-seed sign consensus strategy to improve the stability of the extracted safety task vector.
- Our experiments demonstrate that the extracted safety task vector can effectively induce misalignment while largely preserving model utility, while also exhibiting transferability across models sharing the same pretrained base.

2 Related Work

LLM Misalignment. Existing works have shown that the safety alignment of open-weight LLMs can be easily compromised through post-training modifications. Finetuning-based attacks demonstrated that even limited harmful fine-tuning can effectively remove the safety alignment of LLMs (Yang et al., 2023; Pelrine et al., 2023; Betley et al., 2025; Russinovich et al., 2026). Abliteration (Arditi et al., 2024) suppresses safety refusal by identifying and ablating a refusal direction from model activations, while Heretic (Weidmann, 2025) improves this method with automated hyperparameter optimization. SSRA (Gong et al., 2025) achieves misalignment by manipulating the model's

semantic representations without using harmful responses as fine-tuning labels. TwinBreak (Krauß et al., 2025) localizes and prunes parameters essential to model safety using paired harmful and harmless prompts. Most closely related to ours, Contrastive Weight Steering (Fierro & Roger, 2026) induces misalignment by contrasting and scaling weight differences derived from models with opposing safety behaviors. However, these attacks are typically constructed for a single target model and need to be reconstructed when the target changes.

Task Arithmetic. Task arithmetic represents task-specific behaviors as parameter differences between models and enables model editing by arithmetically combining task vectors. Ilharco et al. (2023) first introduced task vectors as directions in weight space and showed that arithmetic operations such as addition, negation or scaling can enhance, remove or modulate model capabilities. Subsequent work has investigated redundancy and sign conflict in task vectors. TIES-Merging (Yadav et al., 2023), DARE (Yu et al., 2024), and FuseChat (Wan et al., 2025) show that a large fraction of task vector parameters are redundant and can be removed with limited impact on task performance. They further demonstrate that resolving sign conflicts can improve the effectiveness of task arithmetic. Building on these observations, NegMerge (Kim et al., 2025) leverages sign consensus across task vectors obtained under different hyperparameters to identify stable task-relevant updates and filter out redundant parameters.

3 Background and Preliminaries

Threat Model. We consider a threat model where an adversary aims to remove the safety alignment of an open-weight target language model θ_{tgt} while preserving its general utility as much as possible. Specifically, the target model is assumed to be derived from a publicly available pretrained base model θ_{pre}. This setting is common on the open-source LLM platforms such as Hugging Face[1], where post-trained models are often released together with publicly accessible pretrained base models such as Llama (Grattafiori et al., 2024) and Qwen (Yang et al., 2025). We assume that θ_{tgt} and θ_{pre} share the same architecture, and an adversary can freely download both checkpoints and therefore has white-box access to their model parameters. Besides, we also assume that the adversary does not use any harmful training responses in the attack. This restriction establishes a more difficult attack setting that the adversary can only induce misalignment by manipulating the safety knowledge already encoded in the model parameters.

Task Arithmetic. Given a shared reference model θ_{ref}, task-specific models are typically obtained by fine-tuning θ_{ref} on different tasks. For a task t, let θ_{t} denote the model fine-tuned from θ_{ref} on task t. The corresponding task vector of t is defined as the element-wise parameter difference $\tau_{\mathrm{ref}\to\mathrm{t}} = \theta_{\mathrm{t}} - \theta_{\mathrm{ref}}$. Task arithmetic (Ilharco et al., 2023) shows that such task vectors can edit model behavior through simple arithmetic operations in the parameter space. Given n task vectors $\{\tau_{\mathrm{ref}\to\mathrm{t}_i}\}_{i=1}^{n}$ with $n \geq 1$, the edited model can be generally expressed as

$$\theta_{\mathrm{new}} = \theta_{\mathrm{ref}} + \sum_i \lambda_i \tau_{\mathrm{ref}\to\mathrm{t}_i}, \tag{1}$$

where a positive λ_i moves the model along the task directions, thereby incorporating the corresponding task behaviors, while a negative λ_i moves the model in the opposite directions and can suppress these behaviors. The magnitude of λ_i controls the strength of the parameter-space modification and is typically tuned to modulate the edit strength.

4 Method

In this section, we first describe the two critical steps of our attack and then provide an overview of the attack pipeline.

4.1 Safety Task Vector Construction

Based on task arithmetic, the adversary can construct a safety task vector τ_{safe}, and then edit the target model by $\theta_{\mathrm{tgt}} - \lambda\tau_{\mathrm{safe}}$ to break the safety alignment of θ_{tgt}. In this part, we first discuss how

[1] https://huggingface.co

to choose the reference model so that the resulting task vector can be transferred across different target models, and then introduce a paired extraction strategy to reduce the interference of general instruction-related updates.

First, we discuss the choice of reference model for constructing the safety task vector. Existing weight steering method (Fierro & Roger, 2026) treats θ_{tgt} as the reference model and constructs safety task vectors by fine-tuning the target model θ_{tgt}. However, this safety direction is tied to the particular target model used as the reference, and is not naturally transferable to other instruction-tuned variants derived from the same pretrained model. The reason is that a task vector is determined not only by the training objective, but also by the reference model where the optimization starts. For example, consider two models θ_{a}, θ_{b} derived from the same pretrained base θ_{pre}, and we extract the safety task vector $\tau_{\mathrm{a}\to\mathrm{safe}}$ from θ_{a}. Since fine-tuning updates depend on their initialization, there is generally no reason to expect $\tau_{\mathrm{a}\to\mathrm{safe}} = \tau_{\mathrm{b}\to\mathrm{safe}}$. Consequently, applying $\tau_{\mathrm{a}\to\mathrm{safe}}$ to θ_{b} does not necessarily recover the safety direction of θ_{b} (Ilharco et al., 2023; Lin et al., 2025). In contrast, models derived from the same pretrained model naturally share θ_{pre} as a common reference point. Therefore, we can construct a safety task vector $\tau_{\mathrm{pre}\to\mathrm{safe}}$ from θ_{pre}, and apply it to any downstream model. Taking θ_a as an example, the resulting model can be written as $\theta_{\mathrm{a}} - \lambda\tau_{\mathrm{pre}\to\mathrm{safe}} = \theta_{\mathrm{pre}} + \tau_{\mathrm{pre}\to\mathrm{a}} - \lambda\tau_{\mathrm{pre}\to\mathrm{safe}}$. This formulation directly follows the task arithmetic operation described in Equation 1, where task vectors from the same reference can be composed through scaled addition or subtraction to combine their corresponding behaviors in a single model.

This observation motivates us to construct the safety task vector from the pretrained model θ_{pre}. A naive approach under our no-harmful-data setting is to fine-tune θ_{pre} on safety data to obtain θ_{safe}, and then construct the safety vector as $\tau_{\mathrm{pre}\to\mathrm{safe}} = \theta_{\mathrm{safe}} - \theta_{\mathrm{pre}}$. However, this vector may also capture substantial parameter changes unrelated to safety alignment. Since θ_{pre} has not yet undergone instruction tuning, fine-tuning it on instruction-formatted safety data introduces not only safety-related behaviors, but also parameter changes associated with general capabilities such as instruction-following and response formatting. Consequently, the resulting vector captures parameter changes associated with both general capabilities and safety alignment, making it difficult to directly characterize the safety-related updates without interference from general instruction learning.

To construct a safety task vector without these general instruction-related components, we propose a paired extraction method that leverages the subtractive property of task arithmetic to subtract the parameter changes induced by general instruction tuning (Ilharco et al., 2023). Specifically, let $\mathcal{D}_{\mathrm{gen}}$ denote the general instruction dataset and $\mathcal{D}_{\mathrm{safe}}$ denote the safety dataset. We finetune θ_{pre} on $\mathcal{D}_{\mathrm{gen}}$ to obtain θ_{gen}, and on the combined dataset $\mathcal{D}_{\mathrm{gen}} \cup \mathcal{D}_{\mathrm{safe}}$ to obtain $\theta_{\mathrm{gen+safe}}$. The corresponding task vectors are then defined as $\tau_{\mathrm{pre}\to\mathrm{gen}} = \theta_{\mathrm{gen}} - \theta_{\mathrm{pre}}$ and $\tau_{\mathrm{pre}\to\mathrm{gen+safe}} = \theta_{\mathrm{gen+safe}} - \theta_{\mathrm{pre}}$. Since both task vectors are obtained from the same reference model θ_{pre} and share the general instruction-tuning data, they are expected to contain a substantial common component associated with general instruction learning. Under task arithmetic, subtracting $\tau_{\mathrm{pre}\to\mathrm{gen}}$ from $\tau_{\mathrm{pre}\to\mathrm{gen+safe}}$ can therefore suppress the directions associated with general instruction tuning while retaining the direction associated with safety alignment. Accordingly, we construct the safety task vector τ_{safe} as:

$$\tau_{\mathrm{safe}} = \tau_{\mathrm{pre}\to\mathrm{gen+safe}} - \tau_{\mathrm{pre}\to\mathrm{gen}} = \theta_{\mathrm{gen+safe}} - \theta_{\mathrm{gen}}. \tag{2}$$

The final misaligned model θ_{mis} can be expressed as:

$$\theta_{\mathrm{mis}} = \theta_{\mathrm{tgt}} - \lambda\tau_{\mathrm{safe}} = \theta_{\mathrm{tgt}} - \lambda(\theta_{\mathrm{gen+safe}} - \theta_{\mathrm{gen}}). \tag{3}$$

To examine whether paired extraction is beneficial when constructing the safety task vector from the pretrained reference, we conduct a toy experiment with Llama-3.2-3B as θ_{pre} and Llama-3.2-3B-Instruct as θ_{tgt}. Specifically, we compare two extraction strategies: (1) direct extraction, which constructs the task vector by fine-tuning the reference model on safety data, and (2) paired extraction, which subtracts the general instruction task vector from the general-plus-safety task vector to suppress the shared general direction. For each strategy, we remove the extracted safety direction from the target model by $\theta_{\mathrm{tgt}} - \lambda\tau_{\mathrm{safe}}$. We then evaluate whether paired extraction produces more effective safety removal while better preserving the utility of the target model. The detailed configurations of this experiment is consistent with those described in Section 5.1.

As shown in Figure 2, paired extraction largely improves the ASR-utility trade-off of safety removal. In particular, for comparable attack success rate, paired extraction better preserves the utility of the target model. This result is consistent with our motivation that direct extraction from a pretrained model can contain instruction-related parameter changes, whereas paired extraction can reduce such interference. Based on this observation, we use paired extraction to construct the safety task vector τ_{safe} throughout the remainder of this work.

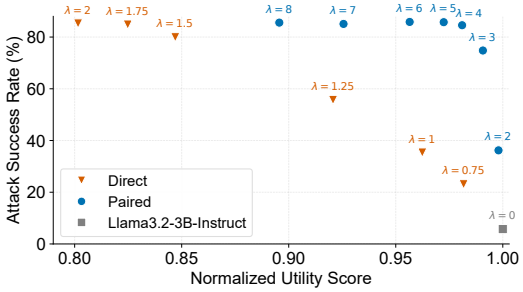

Figure 2: The ASR–utility trade-off under different safety task vector extraction strategies. In the figure, 'Direct' and 'Paired' denote two safety direction extraction strategies, while Llama3.2-3B-Instruct denotes the original target model before safety removal. Labels next to each point indicate the corresponding scaling coefficient λ.

4.2 Task Vector Denoising

However, a safety task vector obtained from a single fine-tuning run may still contain redundant or noisy parameter changes caused by training stochasticity (Yadav et al., 2023; Yu et al., 2024), which may interfere with model utility during safety task vector subtraction. NegMerge (Kim et al., 2025) shows that sign-consistent elements across task vectors derived from models fine-tuned on the same data under different training configurations are more likely to capture stable task-related information. Following the sign-consensus principle of NegMerge, we construct N safety task vectors $\{\tau_{\text{safe}}^{(i)}\}_{i=1}^{N}$ with different training seeds while keeping all other training settings identical. Subsequently, we retain only parameter positions whose signs are consistent across all N task vectors and set the remaining positions to zero. For each retained position, we average its values across the N task vectors to obtain the final denoised safety task vector $\hat{\tau}_{\text{safe}}$. This allows us to suppress training-specific variations and obtain a more stable safety direction. The ablation study of N in task vector denoising is shown in Section 5.4.

4.3 Overall Attack Pipeline

Combining the two steps above, we summarize the overall pipeline of our method in Algorithm 1. Given the pretrained model θ_{pre} and the target model θ_{tgt}, we construct a stable safety task vector through multiple paired fine-tuning runs and use it to remove safety alignment from the target model. We first prepare a general instruction dataset $\mathcal{D}_{\text{gen}}$ and a safety dataset $\mathcal{D}_{\text{safe}}$, and then proceed with the following three steps:

- **Safety Task Vector Extraction.** We perform N extraction runs with different training seeds. In each run, we fine-tune θ_{pre} on $\mathcal{D}_{\text{gen}}$ and $\mathcal{D}_{\text{gen}} \cup \mathcal{D}_{\text{safe}}$ to obtain $\theta_{\text{gen}}^{(i)}$ and $\theta_{\text{gen+safe}}^{(i)}$, and construct safety task vectors as $\tau_{\text{safe}}^{(i)} = \theta_{\text{gen+safe}}^{(i)} - \theta_{\text{gen}}^{(i)}$.
- **Task Vector Denoising.** Next, for each parameter position in the task vector, we retain it only when all N safety task vectors have the same sign and average the retained values across all runs. Otherwise, the position is set to zero. The denoised safety task vector is denoted as $\hat{\tau}_{\text{safe}}$.
- **Safety Removal.** Finally, we subtract the denoised safety task vector from the target model by $\theta_{\text{mis}} = \theta_{\text{tgt}} - \lambda\hat{\tau}_{\text{safe}}$, where λ is a scaling hyperparameter that controls the magnitude of the removal.

5 Experiments

5.1 Experiment Settings

Models. We select model families with both pretrained base models and fine-tuned variants publicly available on Hugging Face. Specifically, we use Llama-3.2-3B, Llama-3.1-8B (Grattafiori et al.,

Table 1: ASR-Utility trade-off across different target models and attack methods. **Bold** indicates the best-performing method for each model and benchmark. Original denotes the unattacked target model. Among the evaluated attack methods, SA, HS, and CWS require harmful training responses, whereas Heretic and our method do not.

Target Model	Method	Utility Benchmarks ↑ (%)					Attack Success Rate ↑ (%)			
		MMLU	HellaSwag	ARC-C	GSM8K	**Utility Score**	AdvBench	SorryBench	StrongReject	**ASR**
Llama-3.2-3B-Instruct	Original	**62.3**	71.6	45.7	70.5	62.5	1.0	13.8	2.6	5.8
	SA	61.3	70.4	43.0	65.7	60.1 (-2.4)	**95.6**	64.4	**93.3**	84.4
	HS	60.8	71.5	45.1	69.4	61.7 (-0.8)	93.8	68.7	90.4	84.3
	CWS	61.5	72.2	**47.6**	68.4	62.4 (-0.1)	94.8	68.4	89.8	84.3
	Heretic	62.1	71.6	45.6	**71.3**	**62.6** (+0.1)	92.5	69.8	87.2	83.2
	Ours	61.9	**71.7**	47.2	66.7	61.9 (-0.6)	95.0	**70.2**	91.4	**85.5**
Llama-3.1-8B-Instruct	Original	**68.4**	79.5	55.9	**77.4**	70.3	5.0	10.9	1.0	5.6
	SA	65.3	77.1	54.2	65.6	65.6 (-4.7)	94.6	68.9	**94.9**	**86.1**
	HS	65.6	76.5	53.3	52.5	62.0 (-8.3)	92.7	67.6	90.7	83.7
	CWS	68.0	**81.6**	**58.4**	77.0	**71.3** (+1.0)	94.0	67.8	92.3	84.7
	Heretic	68.3	79.4	54.9	**77.4**	70.0 (-0.3)	**95.8**	**70.7**	90.4	85.6
	Ours	66.0	78.8	54.7	74.5	68.5 (-1.8)	95.2	69.3	89.5	84.7
Qwen3-4B	Original	68.2	68.5	53.9	76.7	66.8	1.2	15.6	6.1	7.6
	SA	**68.6**	69.3	54.6	71.6	66.0 (-0.8)	96.9	69.1	**95.2**	87.1
	HS	**68.6**	70.1	55.6	73.3	66.9 (+0.1)	85.6	68.2	86.6	80.1
	CWS	68.1	**72.3**	**57.3**	**80.7**	**69.6** (+2.8)	95.6	72.7	93.9	87.4
	Heretic	68.5	68.6	53.4	74.7	66.3 (-0.5)	**97.5**	**73.1**	93.0	**87.9**
	Ours	67.0	67.1	53.7	77.3	66.3 (-0.5)	94.6	70.7	92.0	85.8
Qwen3-8B	Original	**73.0**	74.9	56.1	77.8	70.4	3.7	19.8	18.8	14.1
	SA	72.6	74.2	57.5	78.5	70.7 (+0.3)	**97.1**	67.1	**93.6**	85.9
	HS	**73.0**	75.8	59.4	81.2	72.4 (+2.0)	58.8	57.8	78.3	65.0
	CWS	72.9	**78.6**	**60.8**	**84.1**	**74.1** (+3.4)	95.6	**71.8**	93.0	**86.8**
	Heretic	72.9	74.9	56.9	77.6	70.6 (+0.2)	55.2	55.3	65.8	58.8
	Ours	72.4	74.6	58.0	76.5	70.4 (+0.0)	91.5	66.0	81.5	79.7
Gemma3-4B-it	Original	**57.6**	74.3	57.3	82.0	**67.8**	4.6	17.1	6.4	9.4
	SA	55.9	72.5	52.9	72.7	63.5 (-4.3)	96.5	64.9	90.4	83.9
	HS	56.8	73.9	52.2	82.9	66.5 (-1.3)	63.7	60.2	76.0	66.6
	CWS	57.3	**78.0**	**59.2**	74.8	67.3 (-0.5)	94.6	69.8	**92.3**	85.6
	Heretic	57.5	74.3	55.7	**83.9**	**67.8** (+0.0)	93.5	66.4	86.9	82.3
	Ours	55.3	73.0	55.4	80.0	65.9 (-1.9)	**96.9**	**70.2**	92.0	**86.4**

2024), Qwen3-4B-Base, Qwen3-8B-Base (Yang et al., 2025), and Gemma-3-4b-pt (Team et al., 2025) as the pretrained models θ_{pre}, and use their official instruction-tuned versions as the target models θ_{tgt}.

Fine-tuning Dataset. We adopt Tulu-3-SFT (Lambert et al., 2024), an open-source post-training dataset that contains both general instruction-following data and safety data. We randomly sample a subset of 40,000 examples while preserving the original proportion of general and safety data, resulting in 35,274 general samples and 4,726 safety samples for constructing the corresponding task vectors.

Fine-tuning Configurations. We perform full-parameter fine-tuning for two epochs with a global batch size of 64 and a learning rate of 5×10^{-6}. We use AdamW as the optimizer with a linear learning-rate scheduler and a warmup ratio of 0.03. The maximum sequence length is set to 2,048 tokens. All task vectors are constructed using models fine-tuned under the same configuration.

Metrics. We evaluate model performance from both utility and misalignment effectiveness. For utility, we report the Utility Score, computed as the average accuracy across well-known general-capability benchmarks, including MMLU (Hendrycks et al., 2020), GSM8K (Cobbe et al., 2021), HellaSwag (Zellers et al., 2019), and ARC-Challenge (Clark et al., 2018) by LM-Evaluation-Harness (Gao et al., 2024). We also report Normalized Utility Score in our figures, which is computed as the ratio of the Utility Score of the misaligned model to that of its original model. For misalignment effectiveness, we report the attack success rate (ASR), defined as the proportion of harmful prompts for which the model generates unsafe responses. Specifically, we generate responses on several harmful-prompt benchmarks, including AdvBench (Zou et al., 2023), StrongREJECT (Souly et al., 2024), and SorryBench (Xie et al., 2025), and use Llama-Guard-3-8B (Grattafiori et al., 2024) to classify whether each response is unsafe. A higher Utility Score indicates better preservation of model utility, while a higher ASR indicates more effective removal of safety alignment.

Hyper-parameters. We use $N = 4$ runs by default and set the scaling coefficient to $\lambda = 10$ for Llama-3 and Gemma-3, and $\lambda = 14$ for Qwen-3 models. The ablation studies of these hyper-parameters are shown in Section 5.4.

5.2 ASR-UTILITY TRADE-OFF

Baselines. We compare our method against existing representative misalignment attacks, including fine-tuning-based methods, Shadow Alignment (SA) (Yang et al., 2023) and Harmful SafeRLHF (HS) (Pelrine et al., 2023), the activation steering-based method Heretic (Weidmann, 2025), and the weight steering-based method Contrastive Weight Steering (CWS) (Fierro & Roger, 2026). Detailed descriptions of these baselines are provided in Appendix A.

Evaluation Results. As shown in Table 1, our method consistently induces substantial misalignment across all five target models while retaining most of their original utility. Among existing baselines, the fine-tuning-based method SA generally achieves high attack success rates, but causes relatively large model utility degradation. HS preserves utility better than SA on most target models, yet its attack effectiveness is less consistent and drops substantially on Qwen3-8B and Gemma3-4B-it. CWS achieves the strongest overall ASR-utility trade-off among the existing baselines, but relies on harmful training responses to construct its weight-steering direction. In contrast, our method and Heretic do not require harmful training responses. Heretic generally preserves model utility well, but its attack effectiveness can vary significantly across target models, with notably lower ASR on Qwen3-8B. Our method maintains consistently high attack effectiveness across different target models while largely preserving their original utility. These results demonstrate that even without harmful training responses, our method can achieve a competitive ASR-utility trade-off across different target models.

5.3 TRANSFERABILITY

To evaluate the transferability of our extracted safety task vector, we apply it to different post-trained models derived from the same pretrained base model and measure its effectiveness in terms of both ASR and utility. Specifically, we select off-the-shelf downstream models obtained through diverse post-training pipelines from the Hugging Face model tree. As shown in Figure 3, taking Llama-3.1-8B as an example, the original ASR varies across these models, reflecting their different post-training objectives and safety levels. After applying the extracted safety task vector, the ASR consistently increases to approximately 80% across all evaluated models. This effect is particularly evident for models with a low original ASR, indicating that the extracted vector can induce substantial safety degradation even when the target model has undergone additional safety alignment or preference optimization. Meanwhile, the normalized utility remains above 0.95 for all models, showing that the increase in harmful responses incurs only a limited loss of model utility. These results suggest that models derived from the same pretrained base can share a safety-related parameter direction despite taking different post-training procedures, enabling the extracted safety task vector to transfer across instruction-tuned (Lambert et al., 2024; Grattafiori et al., 2024), domain-specialized (He et al., 2025; Shi et al., 2024; Abo El-Enen et al., 2026), preference-optimized (Lambert et al., 2024), and distilled variants (Guo et al., 2025). The transferability results for additional models are provided in Appendix D. Details of the used models are provided in Appendix E.

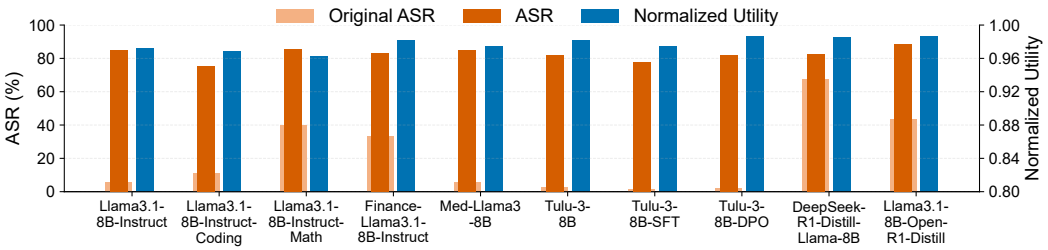

Figure 3: The ASR-utility trade-off of different post-trained models derived from Llama-3.1-8B. The original ASR denotes the ASR before applying the extracted safety task vector.

5.4 Ablation Study

We conduct ablation studies to investigate the effects of hyperparameters in our attack. We use Llama-3.2-3B as the pretrained base model θ_{pre} and Llama-3.2-3B-Instruct as the target model θ_{tgt} in all ablation experiments.

Effect of the Scaling Coefficient λ and Number of Runs N. Figure 4 studies the effect of the scaling coefficient λ and the number of training runs N on the trade-off between attack effectiveness and model utility. For a fixed N, increasing λ first improves the attack effectiveness, with the ASR increasing to a high level and then remaining nearly unchanged. In contrast, the normalized utility score continues to decrease as λ increases. Therefore, the scaling coefficient λ needs to be tuned to get an ideal ASR-utility trade-off. A small λ may not have an ideal ASR, while an overly large λ may cause more utility degradation. This is consistent with the results in task arithmetic (Ilharco et al., 2023; Fierro & Roger, 2026). Moreover, increasing N substantially improves the ASR-utility trade-off. With $N = 1$, achieving an ASR above 85% requires a relatively large sacrifice in utility, with the normalized utility score dropping below 0.98. In contrast, $N = 4$ reaches a comparable ASR while retaining a normalized utility score of around 0.99, and $N = 10$ further preserves the utility around 0.999 at a similar attack effectiveness. Meanwhile, the best λ becomes larger as N increases. This is because fewer parameter updates satisfy the consistency criterion as N increases, making fewer safety-related parameter updated. As a result, the task vector produces a smaller overall safety-related change under the same λ. Therefore, stronger scaling is required to induce sufficient misalignment. However, the retained updates are more consistent across runs, which is more safety-related and reduces unnecessary interference with general capabilities, thereby leading to better utility preservation. Overall, these results demonstrate that using multiple runs in task vector denoising provides a more favorable balance between ASR and model utility, while λ can control the strength of the edit.

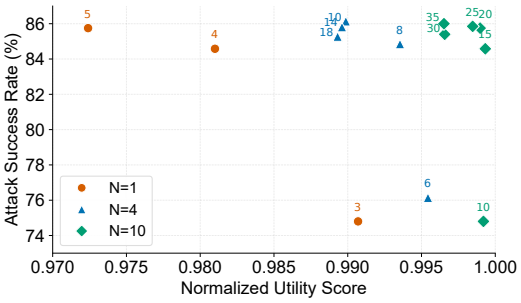

Figure 4: The ASR–utility trade-off under different scaling coefficients λ and number of runs N. Labels next to each point indicate the corresponding scaling coefficient λ.

5.5 Understanding the Extracted Safety Task Vector

Task Vector Similarity with the General Instruction Direction. Ilharco et al. (2023) demonstrated that the task vector for semantically similar tasks have high cosine similarity in weight space. To examine whether paired extraction can suppress general instruction components, we select another 35,274 non-overlapping general instruction samples from the Tulu-3-SFT dataset to get a general instruction task vector τ'_{gen} and measure the cosine similarity between τ'_{gen} and task vectors produced in Section 4. As shown in Figure 5, task vectors trained with general instruction data (i.e., τ_{gen} and $\tau_{\text{gen+safe}}$) have a high cosine similarity above 0.5. The task vector trained directly with safety data $\tau_{\text{safe(Direct)}}$ has a relatively high cosine similarity of 0.282, showing that directly using safety data does capture parameter changes related to instruction following. In contrast, $\tau_{\text{safe(Paired)}}$ has a lower cosine similarity of 0.109, showing that paired extraction can suppress the general instruction component in the safety task vector. Besides, the safety task vector after denoising $\hat{\tau}_{\text{safe(Paired)}}$ have nearly identical cosine similarity as $\tau_{\text{safe(Paired)}}$, demonstrating that the denoising process does not change the semantic meaning related with the general direction.



Figure 5: The cosine similarity between task vectors and the general instruction vector. All task vectors use θ_{pre} as the reference model.

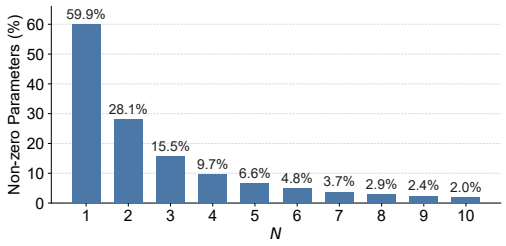

Figure 6: The non-zero parameter ratio of the extracted safety task vector under different number of runs.

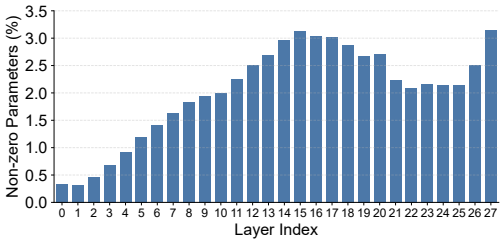

Figure 7: The non-zero parameter ratio of the extracted safety task vector with $N = 10$ in different layers of the Llama-3.2-3B model.

Table 2: The non-zero parameter ratio of the safety vector across different parameter types in Llama3.2-3B model.

Parameter Type	**Attention**				**MLP**			**Embedding**	**Norm**
	Query	Key	Value	Output	Gate	Up	Down		
Non-zero Ratio	1.333%	1.113%	2.044%	2.083%	2.085%	2.091%	2.292%	1.454%	0.001%

Sparsity of the Extracted Safety Task Vector. We further examine the sparsity induced by the sign-consensus process. Figure 6 reports the proportion of non-zero parameters retained when the safety task vector is constructed from different numbers of runs. As N increases from 1 to 10, the retained parameter ratio decreases from 59.9% to 2.0%. In particular, only 9.7% of the parameters remain under the default setting of $N = 4$. This indicates that many parameter updates are not sign-consistent across runs and are therefore removed by the consensus operation. Therefore, transferable safety-related changes can be concentrated in a small subset of parameters.

Distribution of the Extracted Safety Task Vector. We further examine the distribution of the extracted safety task vector with $N = 10$ across transformer layers and parameter types. As shown in Figure 7, the proportion of non-zero parameters generally increases from the shallow layers to the middle layers. Although the proportion decreases in the subsequent layers, it remains higher than that in the shallow layers and rises again in the final layer. Therefore, the transferable safety-related changes are more densely distributed in the middle and later transformer layers. Table 2 further shows how the non-zero parameters are distributed across parameter types. In attention layers, the value and output projections exhibit higher non-zero rates than the query and key projections. The MLP projections consistently retain relatively high proportions of parameters, whereas the embedding parameters have a non-zero rate of 1.454%. In contrast, the non-zero rate of normalization parameters is only 0.001%. These results show that the extracted safety task vector is not uniformly distributed in the model.

6 Conclusion

In this paper, we propose a task arithmetic-based framework for extracting transferable safety directions in parameter space and constructing misaligned LLMs. By anchoring the extraction process to a shared pretrained model and removing general instruction-following components through paired task vector differencing, our approach enables a single safety direction to be reused across multiple models derived from the same pretrained base model. Combined with multi-seed sign consensus, the extracted direction effectively weakens safety alignment while largely preserving general model utility. Extensive experiments demonstrate that the extracted safety task vectors have a competitive ASR-utility trade-off and exhibit strong transferability across diverse post-trained models. Overall, our study provides a new perspective for studying LLM safety alignment in parameter space. We hope our findings can inspire further investigation into how safety behaviors are encoded in LLM weights and contribute to the development of more robust safety alignment techniques.

AI USE STATEMENT

In this work, we used generative AI tools for improving manuscript readability, formatting assistance, finding related works, and code generation. We have reviewed all AI-assisted work. In particular, all AI-assisted code was manually inspected and tested for correctness before being used in experiments, and all AI-assisted writing and literature suggestions were carefully reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Although the proposed approach can induce misalignment in open-weight LLMs, our goal is to improve the understanding of LLM safety mechanisms and facilitate the development of more robust alignment techniques. We do not intend to promote the deployment of unsafe models or enable harmful applications. To reduce potential risks, we will not release any misaligned models or safety task vectors that can be directly used to compromise the safety alignment of publicly available models.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of our method, experimental settings, and evaluation configurations in Section 5.1, Appendix A, and Appendix B. The code of our method is provided in the supplementary material for reproducibility.

REFERENCES

Mohamed Ahmed Abo El-Enen, Sally S. Ismail, and Taymoor Mohamed Nazmy. Med-llama3: Advancing medical question-answering through parameter-efficient fine-tuning of large language models. *Applied Sciences*, 16(12):6158, 2026. doi: 10.3390/app16126158. URL https://www.mdpi.com/2076-3417/16/12/6158.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083, 2024.

Jan Betley, Daniel Chee Hian Tan, Niels Warncke, Anna Sztyber-Betley, Xuchan Bao, Martin Soto, Nathan Labenz, and Owain Evans. Emergent misalignment: Narrow finetuning can produce broadly misaligned llms. In *International Conference on Machine Learning*, pp. 4043–4068. PMLR, 2025.

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

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

Constanza Fierro and Fabien Roger. Steering language models with weight arithmetic. In *International Conference on Learning Representations*, volume 2026, pp. 138122–138169, 2026.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Yichen Gong, Delong Ran, Xinlei He, Tianshuo Cong, Anyu Wang, and Xiaoyun Wang. Safety misalignment against large language models. In *NDSS*, 2025.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized LLMs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=rw50iUoyLu.

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

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023.

Hyo Seo Kim, Dongyoon Han, and Junsuk Choe. Negmerge: Sign-consensual weight merging for machine unlearning. In *Forty-second International Conference on Machine Learning*, 2025.

Anton Korznikov, Andrey Galichin, Alexey Dontsov, Oleg Y Rogov, Ivan Oseledets, and Elena Tutubalina. The rogue scalpel: Activation steering compromises llm safety. *arXiv preprint arXiv:2509.22067*, 2025.

Torsten Krauß, Hamid Dashtbani, and Alexandra Dmitrienko. {TwinBreak}: Jailbreaking {LLM} security alignments based on twin prompts. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 2343–2362, 2025.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Pin-Jie Lin, Rishab Balasubramanian, Fengyuan Liu, Nikhil Kandpal, and Tu Vu. Efficient model development through fine-tuning transfer. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2617–2636, 2025.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. *Advances in Neural Information Processing Systems*, 36:66727–66754, 2023.

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.

Kellin Pelrine, Mohammad Taufeeque, Michał Zając, Euan McLean, and Adam Gleave. Exploiting novel gpt-4 apis. *arXiv preprint arXiv:2312.14302*, 2023.

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*, 36:53728–53741, 2023.

Mark Russinovich, Yanan Cai, Keegan Hines, Giorgio Severi, Blake Bullwinkel, and Ahmed Salem. Grp-obliteration: Unaligning llms with a single unlabeled prompt. *arXiv preprint arXiv:2602.06258*, 2026.

Xiaofeng Shi, Lulu Zhao, Hua Zhou, Donglin Hao, and Yonghua Lin. Industryinstruction, 2024. URL https://huggingface.co/datasets/BAAI/IndustryInstruction.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, et al. A strongreject for empty jailbreaks. *Advances in Neural Information Processing Systems*, 37:125416–125440, 2024.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. Fusechat: Knowledge fusion of chat models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21618–21642, 2025.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Philipp Emanuel Weidmann. Heretic: Fully automatic censorship removal for language models. `https://github.com/p-e-w/heretic`, 2025.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, et al. Sorry-bench: Systematically evaluating large language model safety refusal. In *International Conference on Learning Representations*, volume 2025, pp. 59937–59973, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, 36:7093–7115, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57755–57775. PMLR, 21–27 Jul 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

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

A Baseline Details

Fine-tuning-based Attack. We directly fine-tune the target model θ_{tgt} on harmful instruction–response pairs, using either the Shadow Alignment dataset (Yang et al., 2023) or the Harmful SafeRLHF (Pelrine et al., 2023) dataset as the harmful data. Specifically, we use their official datasets and conduct full-parameter fine-tuning for one epoch with a global batch size of 4 and a learning rate of 1×10^{-5}.

Heretic (Weidmann, 2025) extends the abliteration-based method of Arditi et al. (2024) with automatic hyperparameter optimization, jointly minimizing refusal behavior and the KL divergence from the original model to produce a misaligned model while preserving as much of the original model's capability as possible. We use their official implementation[2] in our experiments.

[2]https://github.com/p-e-w/heretic

Contrastive Weight Steering (Fierro & Roger, 2026) constructs a behavioral direction from the weight difference between models fine-tuned toward safe and unsafe behaviors and applies this direction to the target model. We use their official configuration[3] in our experiments.

B ALGORITHM FOR OUR ATTACK

Algorithm 1 The Overall Attack Pipeline

Input: Pretrained model θ_{pre}, target model θ_{tgt}, general instruction dataset $\mathcal{D}_{\text{gen}}$, safety dataset $\mathcal{D}_{\text{safe}}$, number of runs N, scaling coefficient λ

Output: Misaligned model θ_{mis}

for $i = 1, \ldots, N$ **do**
  Set training seed $s_i = i$
  $\theta^{(i)}_{\text{gen}} \leftarrow \text{FT}(\theta_{\text{pre}}, \mathcal{D}_{\text{gen}}; s_i)$
  $\theta^{(i)}_{\text{gen+safe}} \leftarrow \text{FT}(\theta_{\text{pre}}, \mathcal{D}_{\text{gen}} \cup \mathcal{D}_{\text{safe}}; s_i)$
  Construct the safety task vector: $\tau^{(i)}_{\text{safe}} \leftarrow \theta^{(i)}_{\text{gen+safe}} - \theta^{(i)}_{\text{gen}}$
end for
Initialize $\hat{\tau}_{\text{safe}} \leftarrow \mathbf{0}$
for each parameter position j **do**
  if $\text{sign}(\tau^{(1)}_{\text{safe},j}) = \cdots = \text{sign}(\tau^{(N)}_{\text{safe},j})$ **then**
    $\hat{\tau}_{\text{safe},j} \leftarrow \frac{1}{N} \sum_{i=1}^{N} \tau^{(i)}_{\text{safe},j}$
  end if
end for
Remove the denoised safety direction from the target model: $\theta_{\text{mis}} \leftarrow \theta_{\text{tgt}} - \lambda \hat{\tau}_{\text{safe}}$
return θ_{mis}

C LIMITATIONS

Requirement of the Pretrained Base Model. Our method requires access to the pretrained base model corresponding to the target model, which may limit its applicability when such a base model is unavailable. However, many widely used open-weight LLM families publicly release both pretrained and instruction-tuned models, making this requirement practical in common open-source settings.

Computational Cost. Our method requires multiple fine-tuning runs to construct the safety task vector, which introduces additional computational overhead compared to existing methods. However, this extraction process only needs to be performed once during task-vector construction, and the resulting safety task vector can be reused across multiple target models sharing the same pretrained base, avoiding repeated extraction for each individual model.

Computational Resource Constraints. Due to hardware and computational resource constraints, we can only evaluate our method on LLMs up to the 8B scale. Therefore, its effectiveness and transferability on substantially larger models remain to be further investigated.

[3] https://github.com/safety-research/weight-steering

D ADDITIONAL RESULTS FOR TRANSFERABILITY

As shown in Figures 8, 9, and 10, we further evaluate the transferability of our method on diverse post-trained models derived from Llama-3.2-3B, Qwen3-4B-Base, and Qwen3-8B-Base. Across different model families and post-training pipelines, applying the extracted safety task vector consistently and largely increases ASR compared with the original models. Meanwhile, the normalized utility generally remains close to that of the original models. These results demonstrate that the extracted safety task vector transfers effectively across downstream variants sharing the same pre-trained base model, despite differences in their post-training data and algorithms. The detail of models are available in Appendix E.

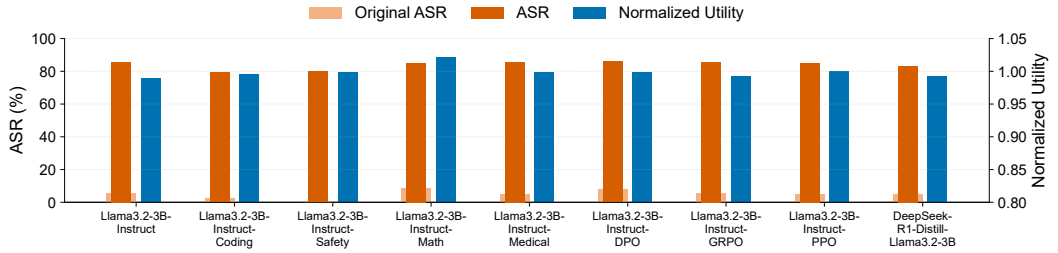

Figure 8: The ASR-utility trade-off of different post-trained models derived from Llama-3.2-3B. The original ASR denotes the ASR before applying the extracted safety task vector.

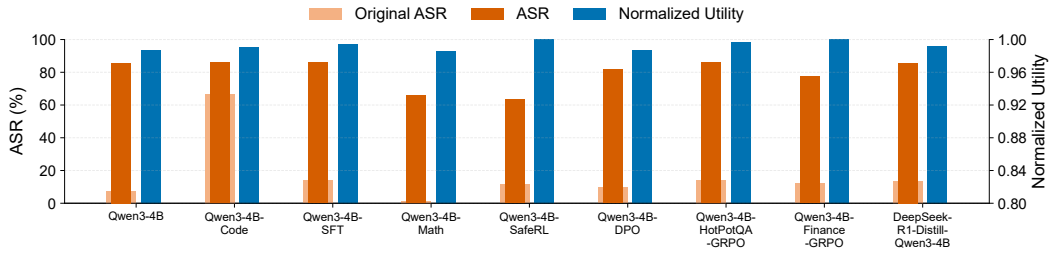

Figure 9: The ASR-utility trade-off of different post-trained models derived from Qwen3-4B-Base. The original ASR denotes the ASR before applying the extracted safety task vector.

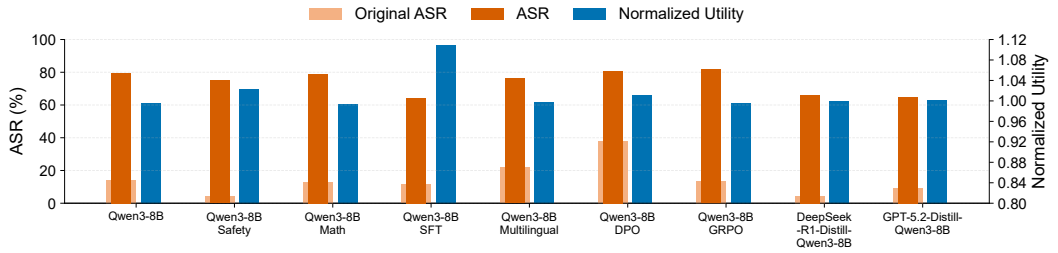

Figure 10: The ASR-utility trade-off of different post-trained models derived from Qwen3-8B-Base. The original ASR denotes the ASR before applying the extracted safety task vector.

E MODELS USED IN TRANSFERABILITY EXPERIMENTS

Table 3: The Hugging Face model paths of the models used in our transferability experiments.

Base Model	Downstream Models
Llama-3.1-8B	meta-llama/Llama-3.1-8B-Instruct MergeBench/Llama-3.1-8B-Instruct_coding MergeBench/Llama-3.1-8B-Instruct_math XiaofengAlg/Finance-llama3_1_8B_instruct MohamedAhmedAE/Llama-3.1-8B-Instruct-Medical-Finetuned-merged allenai/Llama-3.1-Tulu-3-8B allenai/Llama-3.1-Tulu-3-8B-SFT allenai/Llama-3.1-Tulu-3-8B-DPO deepseek-ai/DeepSeek-R1-Distill-Llama-8B asas-ai/Llama-3.1-8B-Instruct-Open-R1-Distill
Llama-3.2-3B	meta-llama/Llama-3.2-3B-Instruct MergeBench/Llama-3.2-3B-Instruct_coding MergeBench/Llama-3.2-3B-Instruct_safety MergeBench/Llama-3.2-3B-Instruct_math MohamedAhmedAE/Llama-3.2-3B-Instruct-Medical-Finetuned-merged MInAlA/Llama-3.2-3B-DPO-merged MInAlA/Llama-3.2-3B-Instruct-GRPO-merged MInAlA/Llama-3.2-3B-Instruct-PPO-merged suayptalha/DeepSeek-R1-Distill-Llama-3B
Qwen3-4B-Base	Qwen/Qwen3-4B prometheus04/qwen3-4b-code-finetuned sergiopaniego/Qwen3-4B-SFT SpyRL/SpyRL-Qwen3-4B-Math Qwen/Qwen3-4B-SafeRL AdamDS/qwen3-security-dpo-4b metacognitive-behavioral-tuning/Qwen3-4B-GRPO Ayansk11/qwen3-4b-financial-sentiment-grpo ertghiu256/deepseek-r1-0528-distilled-qwen3
Qwen3-8B-Base	Qwen/Qwen3-8B qihoo360/TinyR1-Safety-8B SpyRL/SpyRL-Qwen3-8B-Math ajb8866/Qwen3-8B-SFT-fully-trained sergiopaniego/Qwen3-8B-SFT shisa-ai/017-qwen3-8b-v2-dpo405b-clr mlxha/Qwen3-8B-grpo-medmcqa deepseek-ai/DeepSeek-R1-0528-Qwen3-8B TeichAI/Qwen3-8B-GPT-5.2-High-Reasoning-Distill