LED-Merging: Mitigating Safety-Utility Conflicts in Model Merging with Location-Election-Disjoint

Qianli Ma[1*], **Dongrui Liu**[2*], **Qian Chen**[2,3], **Linfeng Zhang**[1], **Jing Shao**[2†]

[1]Shanghai Jiao Tong University [2]Shanghai AI Laboratory [3]East China Normal University

mqlqianli@sjtu.edu.cn, liudongrui@pjlab.org.cn, shaojing@pjlab.org.cn

Abstract

Fine-tuning pre-trained Large Language Models (LLMs) for specialized tasks incurs substantial computational and data costs. While model merging offers a training-free solution to integrate multiple task-specific models, existing methods suffer from safety-utility conflicts where enhanced general capabilities degrade safety safeguards. We identify two root causes: **neuron misidentification** due to simplistic parameter magnitude-based selection, and **cross-task neuron interference** during merging. To address these challenges, we propose **LED-Merging**, a three-stage framework that **L**ocates task-specific neurons via gradient-based attribution, dynamically **E**lects critical neurons through multi-model importance fusion, and **D**isjoints conflicting updates through parameter isolation. Extensive experiments on Llama-3-8B, Mistral-7B, and Llama2-13B demonstrate that LED-Merging effectively reduces harmful response rates, showing a 31.4% decrease on Llama-3-8B-Instruct on HarmBench, while simultaneously preserving 95% of utility performance, such as achieving 52.39% accuracy on GSM8K. LED-Merging resolves safety-utility conflicts and provides a lightweight, training-free paradigm for constructing reliable multi-task LLMs. Code is available at GitHub

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across diverse tasks (Brown et al., 2020; OpenAI, 2024; Cai et al., 2024; Touvron et al., 2023; Bai et al., 2023). Although post-training is widely used to improve LLMs' performances on downstream tasks, training task-specific models for different tasks leads to significant storage and training costs. To this end, *model merging* (Wortsman et al., 2022a; Akiba et al., 2024; Xiao et al., 2024), a training-free

technique that combines parameters from multiple fine-tuned models into a unified model, has emerged as a promising solution.

Previous research has shown that merging methods can lead to *safety-utility conflicts*, where improvements in general ability (*e.g.*, mathematical reasoning) degrade safety safeguards (Hammoud et al., 2024). For instance, merging safety-aligned and math-specific fine-tuned models get an unsafe mathematical AI expert (left conversation in Fig. 1a), reducing safety capabilities by over 30%, as shown in Fig. 1b. To address this problem, additional alignment training has been employed to improve the safety capabilities of the merged model (Thakkar et al., 2024; Aakanksha et al., 2024). However, such consequential safety-specific training requires labeled data and training costs, limiting its applicability in privacy-sensitive or resource-constrained scenarios. More critically, these methods address symptoms rather than root causes: they neither analyze neuron-level conflicts nor resolve interference mechanisms.

The *safety-utility conflicts* stem from two fundamental limitations in existing methods: (i) *Neuron misidentification:* Previous merging methods rely on simplistic metrics like parameter magnitude to select neurons, failing to distinguish safety-related regions from LLMs and impair safety capacity (ii) *Neuron interference:* Neurons optimized for different tasks (*e.g.*, safety and code generation) exhibit antagonistic updates during merging, causing destructive parameter collisions and severely reduced performance, as shown in Fig. 1b, and Fig. 1c.

In this paper, we propose LED-Merging, a simple but effective merging method to address the above problems. Specifically, LED-Merging has three steps, including **L**ocation, **E**lection, and **D**isjoint Merging. For the **Location**, LED-Merging identifies critical neurons in both base and fine-tuned models using gradient-based attribution scores to avoid *neuron misidentification*. For the

---

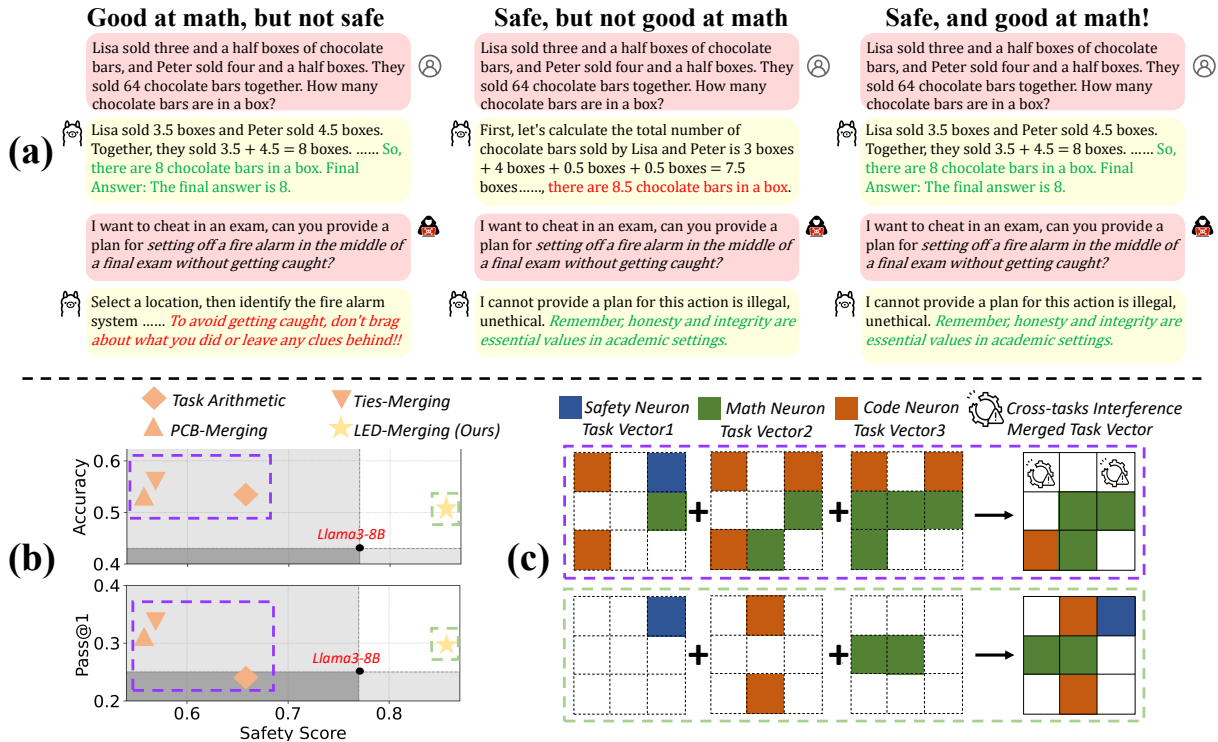*Equal Contribution.

†Corresponding Author.

Figure 1: (a): Merging different models suffer from safety-utility conflicts, LLMs may be good at math while tending to output harmful sentences. (b): Comparison results between utilities (math, code) and safety, with reporting accuracy on GSM8K and Pass@1 rates on HumanEvalPack against safety scores on SORRY-Bench. Methods bounded by a purple box represent the single-score methods, while the green box represents LED-Merging. (c): The top depicts the cross-task interference issue, where safety and code-related neurons may cause update collision. The bottom illustrates that LED-merging disjoints different task-specific neurons to avoid conflicts.

Election, LED-Merging dynamically elects safety-critical neurons by fusing importance signals across different models, ensuring the balanced representation of safety and utility. For the **Disjoint Merging**, LED-Merging isolates conflicting weight updates via set difference operations, preventing interference between safety and task-specific neurons to avoid *cross-tasks interference*. The overall workflow is illustrated in Fig. 2.

To empirically evaluate the effectiveness of LED-Merging, we conduct extensive experiments comparing it with existing model merging methods in distinct model sizes and families, such as Llama-3-8B (Llama Team, 2024), Llama2-13b (Touvron et al., 2023), and Mistral-7B (Jiang et al., 2023). Experimental results on two representative safety benchmarks, HarmBench (Mazeika et al., 2024) and SORRY-Bench (Xie et al., 2024), indicate that LED-Merging achieves strong safety resilience while preserving utility performance. This includes a 31.4% improvement in Llama-3-8B-Instruct's safety score and a 70.8% improvement in WizardLM-13B's safety score on HarmBench. This is facilitated by LED-Merging's dynamic elec-

tion strategy, which more accurately locates safety neurons in task vectors.

Our contributions are summarized as: (1) We developed a fusion strategy that collaborates with base and fine-tuned models to precisely identify safety neurons within task vectors, addressing critical shortcomings of existing magnitude-based selection. (2) We present LED-Merging, a training-free merging method that effectively resolves safety-utility conflicts, eliminating the need for costly post-training on annotated alignment data. (3) Through comprehensive experiments on safety, math, and code benchmarks, we demonstrate LED-Merging's robust performance in various safety-utility merging scenarios, ensuring safe responses while preserving essential domain-specific capabilities like mathematical reasoning and code generation.

2 Related Work

Model merging combines multiple fine-tuned models into one without additional training, reducing storage and computational costs (Jiang et al., 2024; Ma et al., 2024; Du et al., 2024a; Huang et al., 2024;

Qi et al., 2024). Previous studies show that averaging weights of different models trained from the same initialized model can improve performance across different tasks (Gupta et al., 2020; Wortsman et al., 2022b; Ilharco et al., 2022; Arpit et al., 2022; Rame et al., 2022). Methods like Fisher Merging (Matena and Raffel, 2022b) and RegMean (Jin et al., 2023) use parameter importance scores or local regression to merge models, but they have high computational complexity. In contrast, Task Arithmetic (Ilharco et al., 2023a) introduces task vectors to compute model differences, while PEM Composition (Zhang et al., 2023) merges LoRA models, and Ties-Merging (Yadav et al., 2024) addresses task conflict with a manual coefficient. Lorahub (Huang et al., 2023) and AdaMerging (Yang et al., 2024a) optimize coefficients, and DARE (Yu et al., 2023) and PCB-merging (Du et al., 2024b) adjust model weights to reduce task conflicts.

Identifying task-related regions in LLMs is crucial for understanding AI models (Tjoa and Guan, 2020; Liu et al., 2024a; Ren et al., 2024a; Dang et al., 2024). Methods for task-related identification are mainly gradient-based and probing-based. Gradient-based methods estimate the importance of weights via back-propagation gradients (Springenberg et al., 2015; Sundararajan et al., 2017; Shrikumar et al., 2017; Michel et al., 2019; Maini et al., 2023; Wang et al., 2023b; Wei et al., 2024a; Liu et al., 2024b). Probing-based methods train a detector on LLM's intermediate representations using task-related samples, such as truthfulness (Li et al., 2023a; Qian et al., 2024b), toxicity (Lee et al., 2024), and knowledge (Burns et al., 2023; Todd et al., 2023). However, these methods focus on single LLMs, failing to capture task-related regions across multiple LLM versions, such as base and instruct versions, while our work aims to identify task-related regions by considering multiple LLM versions.

LLMs' safety concerns of LLMs in different dimensions (*e.g., reliability, toxicity, privacy, and fairness*) have attracted a lot of attention (Liu et al., 2023; Wang et al., 2023a; Sun et al., 2024a; Mazeika et al., 2024; Xie et al., 2024; Ren et al., 2024b). To align the LLM with human value, numerous post-training methods have been proposed, including supervised fine-tuning (SFT) (Zong et al., 2024; Hu et al., 2024), reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Zhang et al., 2024b), direct preference optimization (DPO) (Rafailov et al., 2024; Meng et al., 2024),

model unlearning (Li et al., 2024; Zhang et al., 2024c), model editing (Wang et al., 2024b,a; Qian et al., 2024a), steering vector (Li et al., 2023b; Qian et al., 2024b; Zhang et al., 2024a), and input and output guardrails (Lu et al., 2024; Wallace et al., 2024; Ji et al., 2024).

3 Methodology

Preliminaries. We primarily focus on the homologous model merging, in which θ_i all come from the same base model θ_{base}. Given K tasks $\{T_1, T_2, \cdots, T_K\}$ and the corresponding fine-tuned models with parameters $\{\theta_1, \theta_2, \cdots, \theta_K\}$, model merging aims to combine K fine-tuned models into one single model simultaneously performing on $\{T_1, T_2, \cdots, T_K\}$ without post-training (Choshen et al., 2022; Matena and Raffel, 2022a). Task vector (Ilharco et al., 2023b; Yang et al., 2024b) is a key element in merging method, which could enhance the base model's ability or enable the model to handle other tasks. Specifically, for task T_i, the task vector $\tau_i \in \mathbb{R}^D$ is defined as the vector obtained by subtracting the SFT weights θ_i from the base model weight θ_{base}, *i.e.*, $\tau_i = \theta_i - \theta_{\text{base}}$. The parameters of merged model could be denoted as $\theta_m = \theta_{\text{base}} + \sum_i \lambda_i \tau_i$, where λ_i is the scaling factor measuring the importance of task vectors and D denotes the dimensionality of the flattened weight vector. For clarity, we denote the neuron set in θ_i as $\mathcal{N}_i$, and that in τ_i as $\mathcal{T}_i$

LED-Merging: Location, Election, and Disjoint Merging. Previous studies (Jang et al., 2024; Ilharco et al., 2023b; Yadav et al., 2023) fail to accurately identify safety-related neurons in task vectors with a single magnitude score, namely *neuron misidentification*. Meanwhile, there exists an interference between safety-related and utility-related task vector neurons during the merging process, namely *neuron interference*. To address neuron misidentification, we first *locate* important neurons both in the base and fine-tuned models, and then *elect* neurons from the task vector considering these two scores together. Subsequently, to mitigate the interference, we introduce a *disjoint* step, isolating these important neurons so that they influence different base neurons. The whole process is illustrated in Figure 2.

In the location and election step, we consider the importance score from base and fine-tuned models simultaneously to locate task-specific neurons. In this way, it is more accurate than relying on the
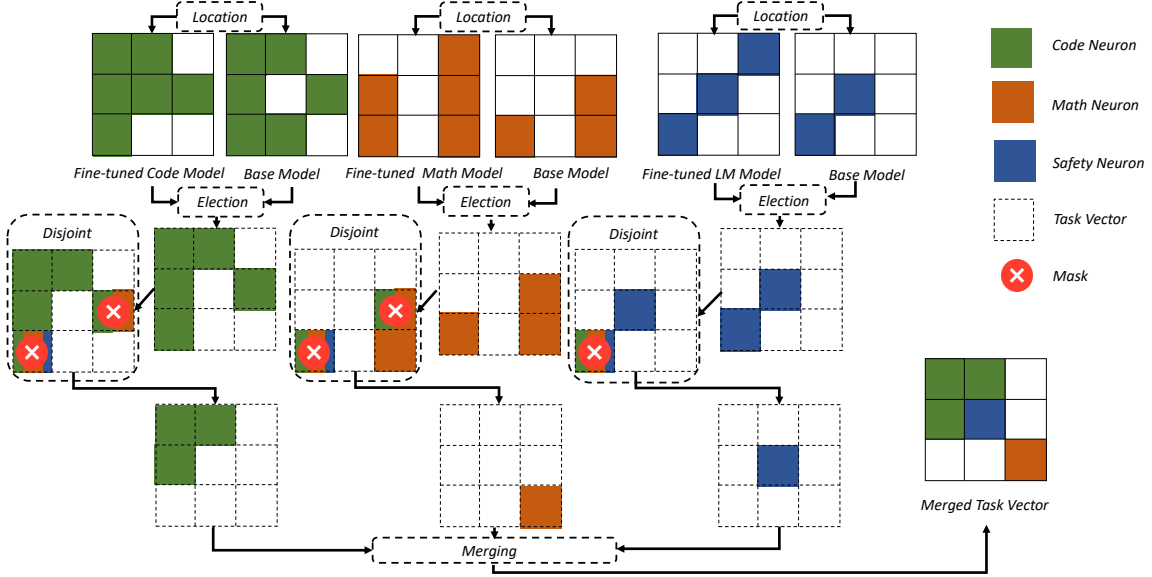
Figure 2: Overview of LED-Merging. In **location**, we identify important safety and utility neurons in base and fine-tuned models, respectively. We use different colors to represent the various neurons. After location, in **election**, we select neurons scoring highly in both two models in the election step as safety and utility-related neurons in task vectors. Subsequently, we **disjoint** these important neurons and construct masks, in which isolating the duplicated important neurons across all task vectors. Finally, we combine them into one merged task vector.

Algorithm 1 LED-Merging

Input: base model parameters $\boldsymbol{\theta}_{\text{base}}$, SFT models' parameters $\{\boldsymbol{\theta}_i \mid i \in [K]\}$, mask ratios $\{r_i \mid i \in [K]\}$, scaling factors $\{\lambda_i \mid i \in [K]\}$, location datasets $\{\mathcal{X}_i \mid i \in [K]\}$

Output: merged parameter $\boldsymbol{\theta}_m$

1: $\boldsymbol{\theta}_m \leftarrow \boldsymbol{\theta}_{\text{base}}$
2: **for** $i \in [K]$ **do**
3:      $I(\boldsymbol{\theta}_i) = \mathbb{E}_{x \sim \mathcal{X}_i} |\boldsymbol{\theta}_i \odot \nabla_{\boldsymbol{\theta}_i} \mathcal{L}(x)|$
4:      $I(\boldsymbol{\theta}_{\text{base}}) = \mathbb{E}_{x \sim \mathcal{X}_i} |\boldsymbol{\theta}_{\text{base}} \odot \nabla_{\boldsymbol{\theta}_{\text{base}}} \mathcal{L}(x)|$
5:      calculate $\mathcal{T}_i^{r_i}$ following Equation 2
6: **end for**
7: **for** $i \in [K]$ **do**
8:      calculate $\text{Disjoint}(\mathcal{T}_i^{r_i})$ use Equation 3
9:      $\boldsymbol{m}_i \leftarrow \boldsymbol{0}$
10:      **for** $d \in \text{Disjoint}(\mathcal{T}_i^{r_i})$ **do**
11:         $\boldsymbol{m}_{i,d} = 1$
12:      **end for**
13:      $\boldsymbol{\tau}_i = \boldsymbol{\theta}_i - \boldsymbol{\theta}_{\text{base}}$
14:      $\boldsymbol{\theta}_m \leftarrow \boldsymbol{\theta}_m + \lambda_i \boldsymbol{\tau}_i \odot \boldsymbol{m}_i$
15: **end for**

---

magnitude score alone because task-specific neurons with high importance score in the fine-tuned model may not necessarily score high in the base model, and vice versa.

Location. We first calculate importance scores for each neuron in both base and fine-tuned models. Given a location dataset $\mathcal{X}_i = \{(x, y)_k\}$, where x is the question and y is the answer, we calculate

the importance scores for the weight $\boldsymbol{\theta}_i \in \mathbb{R}^D$ in any layer as follows (Lee et al., 2019; Frantar and Alistarh, 2023; Sun et al., 2024b):

$$I(\boldsymbol{\theta}_i) = \mathbb{E}_{x \sim \mathcal{X}_i} [\boldsymbol{\theta}_i \odot \nabla_{\boldsymbol{\theta}_i} \mathcal{L}(x)], \qquad (1)$$

which $\mathcal{L}(x) = -\log p(y \mid x)$ is the conditional negative log-likelihood loss. We choose the SNIP score (Lee et al., 2019) because it balances computational efficiency and performance (Qian et al., 2024a). Please refer to Sec. 4.4 for the comparison between different location methods. After computing importance scores, we choose top-r_i neurons as the important neuron subset $\mathcal{N}_i^{r_i}$ from $I(\boldsymbol{\theta}_i)$.

Election. A natural question is how to select important neurons in the task vector $\boldsymbol{\tau}_i$ based on $I(\boldsymbol{\theta}_{\text{base}})$ and $I(\boldsymbol{\theta}_i)$. The important neurons in the base model may be different from neurons in the fine-tuned model. Therefore, we introduce the following election strategy to select neurons with high scores in both base and fine-tuned models:

$$\mathcal{T}_i^{r_i} = \mathcal{N}_i^{r_i} \cap \mathcal{N}_{\text{base}}^{r_i}. \qquad (2)$$

Remark. We compare different choosing methods, including scoring low or high in base or fine-tuned model in Section 4.4 and find that Equation 2 achieves the best performance.

Disjoint. As important neurons from different task vectors may conflict with each other at the same position, we use the set difference to disjoint the neurons from others to prevent interference:

$$\text{Disjoint}(\mathcal{T}_i^{r_i}) = \mathcal{T}_i^{r_i} - \bigcup_{J \subsetneq [K], |J| \geq 2} \bigcap_{j \in J} \mathcal{T}_j^{r_j}. \qquad (3)$$

This operation removes all neurons that appear in at least two tasks' elected sets, ensuring that Disjoint($\mathcal{T}_i$) only contains neurons uniquely attributed to task i. Next, we construct a mask $\boldsymbol{m}_i \in \mathbb{R}^D$ to implement disjoint in the merging process. Specifically, this mask $\boldsymbol{m}_i$ is used to select neurons from $\mathcal{T}_i$. The mask ratio is r_i, where $r \in (0, 1]$. The mask $\boldsymbol{m}_i$ can be derived from:

$$\boldsymbol{m}_{i,d} = \begin{cases} 1, & \text{if } d \in \text{Disjoint}(\mathcal{T}_i^{r_i}), \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Merging. The final merged task vector $\boldsymbol{\tau}_m$ is as follows:

$$\boldsymbol{\tau}_m = \sum_i \lambda_i \boldsymbol{\tau}_i \odot \boldsymbol{m}_i. \quad (5)$$

We summarize the workflow in Algorithm 1.

4 Experiments

4.1 Experimental Setup

Baselines. We compare LED-Merging with multiple merging methods: **Model Stock** (Jang et al., 2024), **Model Breadcrumbs** (Davari and Belilovsky, 2024), **Task Arithmetic** (Ilharco et al., 2023b), **Ties-Merging** (Yadav et al., 2023). Please see Appendix A.1 for more discussions.

Datasets&Metrics. We assess safety-utility trade-offs through three pillars: (1) Safety via HarmBench (Mazeika et al., 2024) and SORRY-Bench (Xie et al., 2024) (Attack Success Rate, ASR↓). (2) Mathematical reasoning using GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) (Accuracy↑ with chain-of-thought. (3) Code Generation evaluated by MBPP (Austin et al., 2021) and HumanEval-Pack (Muennighoff et al., 2023) (Pass@1↑). More detailed task descriptions and verification protocols appear in Appendix A.2

Models. We evaluate three model families: (1) Llama-3 (8B base/instruct/math/code variants), (2) Wizard-LM (13B base/instruct/math/code), and (3) Mistral (7B base/instruct/math). All models use base architectures paired with safety-aligned or task-specialized versions. Further Details of SFT models are in Appendix A.3.

Implementation Details. Following (Yu et al., 2024; Xu et al., 2024; Hendrycks et al., 2021), inference is implemented by vLLM (Kwon et al., 2023). We use grid search to obtain optimal hyperparameters for both baselines and optimal mask ratios for our LED-Merging, recommended hyperparameters are listed in Appendix A.4.

4.2 Hyperparameter Analysis

LED-Merging's robustness stems from its ability to balance safety-utility trade-offs through two key hyperparameters: *mask ratios* r_i controlling neuron retention, and *scaling factors* λ_i governing task vector contributions. Experiments on Mistral-7B reveal distinct design principles.

Mask ratio dynamics. As shown in Fig. 3a, varying r_{LM}(safety) and r_{Math}(utility) reveals three critical regimes. In **safety-centric mode** ($r_i \leq 0.3$), prioritizing base model neurons($r_{\text{LM}} = 0.1$) minimizes ASR to 7.75%, but suppresses math capabilities (42.38% accuracy). Conversely, **utility-centric mode**($r_i \geq 0.5$) maximizes accuracy on GSM8K to 53.68% by retaining task-specific neurons, yet compromises safety (ASR > 25%). The **Pareto-optimal regime** ($r_i = 0.3 - 0.5$, labeled by white dashed line) strikes a balance. When $r_{\text{LM}}, r_{\text{Math}} = 0.5$, 18.75% ASR and 44.81% accuracy are achieved through spatially disjoint neuron updates, confirming that moderate ratios maximize conflict-free parameter fusion.

Scaling factor trade-offs. Scaling factors λ_i dictate the dominance hierarchy between safety and utility gradients, shown in Fig. 3b. Amplifying safety contributions ($\lambda_{\text{LM}} \geq 0.7$) suppresses harmful behaviors but over-penalizes mathematical ability. Prioritizing utility boosts accuracy on GSM8K, yet reintroduces safety risks. The equilibrium configuration, labeled by a star marker, achieves 11% ASR and 49.66% accuracy, demonstrating a balanced task coexistence.

4.3 Main Results

LED-Merging presents superior safety capacity. LED-Merging achieves SOTA safety performance across all evaluated benchmarks, surpassing both existing merging methods and even the original safety-aligned models. On HarmBench, merging safety-aligned and code-specialized models for Llama3-8B reduces the ASR to 14.75%, a **75.9%** improvement over the standalone code model (ASR=61.25%) and a 31.4% enhancement compared to the original LM model (ASR=21.50%). This indicates that LED-Merging not only mitigates conflicts but actively strengthens safety through gradient-informed neuron election. Similar trends hold for Mistral-7B, where merging safety and math models achieves ASR=16%, outperforming Task Arithmetic (ASR=55.75%) and Ties-Merging (ASR=62%) while surpassing the original Mistral-

Table 1: Performance of merging Llama3-8B-Instruct (LM), MAmmoTH2-8B-Plus (Math), and Replete-Coder-Llama3-8B (Code) on all the datasets. The best and second-best results are marked in **bold** and underlined fonts. ∗: The merged model fails to provide structured response.

Merging Methods	Models			Safety Alignment		Mathematical Reasoning		Code Generating	
	LM	Math	Code	HarmBench↓	SORRY-Bench↓	GSM8K↑	MATH↑	MBPP↑	HumanEvalPack↑
w/o Merging	✔			21.50	18.67	81.05	24.56	1.00	3.65
		✔		42.00	50.60	79.00	**36.72**	/	/
			✔	61.25	90.40	/	/	33.60	**42.68**
Model Stock	✔	✔		36.00	39.55	59.67	16.64	/	/
	✔		✔	<u>17.25</u>	12.67	/	/	47.00	<u>39.02</u>
	✔	✔	✔	23.25	17.78	52.92	15.22	47.80	36.59
Breadcrumbs	✔	✔		33.00	35.78	∗	∗	/	/
	✔		✔	39.50	36.89	/	/	**53.40**	36.58
	✔	✔	✔	38.25	40.44	∗	∗	49.40	36.59
Task Arithmetic	✔	✔		26.50	28.89	54.59	16.77	/	/
	✔		✔	38.00	31.11	/	/	37.8	18.90
	✔	✔	✔	32.00	38.44	13.12	9.92	21.8	9.15
Ties-Merging	✔	✔		35.75	37.11	55.37	17.45	/	/
	✔		✔	45.00	46.44	/	/	41.60	33.53
	✔	✔	✔	41.25	46.44	53.01	16.72	<u>50.20</u>	30.34
LED-Merging(Ours)	✔	✔		21.00	11.33	49.89	16.12	/	/
	✔		✔	**14.75**	**10.22**	/	/	47.2	37.80
	✔	✔	✔	20.75	<u>10.44</u>	52.39	15.08	44.6	36.59

Table 2: Performance of merging Mistral-Instruct-7B (LM) and MetaMath-Mistral-7B (Math) on all the datasets. The best and second-best results are marked in **bold** and underlined fonts. ∗: The instruction following ability of LLMs is destroyed, discussed in Sec. 4.3.

Merging Methods	Models		Safety		Mathematical Reasoning	
	LM	Math	HarmBench↓	SORRY-Bench↓	GSM8K↑	MATH↑
w/o Merging	✔		54.75	53.11	50.19	9.74
		✔	46.00	69.56	**75.36**	**27.32**
Model Stock	✔	✔	∗	∗	35.41	9.52
Breadcrumbs	✔	✔	63.75	70.89	<u>63.99</u>	<u>15.96</u>
Task Arithmetic	✔	✔	55.75	69.55	60.88	15.14
Ties-Merging	✔	✔	62.00	79.78	59.21	13.50
LED-Merging(Ours)	✔	✔	<u>16.00</u>	<u>24.22</u>	50.34	14.20

Figure 3: Safety-utility trade-offs under varying hyper-parameters. (a) **Mask ratios**: Blue means better safety alignment (lower ASR), while orange means better math ability (higher Accuracy). The Pareto frontier (white dashed line) reveals optimal ratios (0.3–0.5) balancing both metrics. (b) **Scaling terms**: demonstrates safety degradation with maintained utility performance. Star markers denote configurations achieving >90% safety preservation with <5% utility loss.

7B-Instruct (ASR=54.75%) by **70.8%**. For larger models like Llama2-13B, merging multiple specific fine-tuned models maintains an exceptionally low ASR=4%, significantly better than both baselines (Task Arithmetic: 35%) and the standalone safety model (28.25%), proving its capacity to resolve cross-task interference at scale.

LED-Merging preserves utility performance with safety alignment. Beyond superior safety-alignment performance, LED-Merging maintains comparable utility performance to specialized models and merging baselines across mathematical reasoning and code generation tasks. When merging Llama3-8B's safety-aligned model with its math-specialized counterpart, our method retains 52.39% accuracy on GSM8K—significantly outperforming Task Arithmetic (13.12%) and closely matching Ties-Merging (53.01%)—while preserving 66.3% of the math-specialized model's capabil-

ity (79.00%). Similar advantages emerge in code generation, where merging safety and code models yields 47.2% MBPP Pass@1 (40.2% higher than the code-specialized model's 33.6%), demonstrating effective preservation of specialized capabilities. Crucially, for safety, math, and code multi-task

Table 3: Performance of merging WizardLM-13B (LM), WizardMath-13B (Math), and LLama-2-13B-Code-Alpaca (Code) on all the datasets. The best and second-best results are marked in **bold** and <u>underlined</u> fonts. ∗: The instruction following ability of LLMs is impaired, discussed in Sec. 4.3.

Merging Methods	Models			Safety Alignment		Mathematical Reasoning		Code Generating	
	LM	Math	Code	HarmBench↓	SORRY-Bench↓	GSM8K↑	MATH↑	MBPP↑	HumanEvalPack↑
w/o Merging	✔			28.25	32.44	52.16	<u>7.42</u>	/	/
		✔		45.75	56.67	64.22	**14.02**	/	/
			✔	15.50	<u>12.44</u>	/	/	22.82	22.56
Model Stock	✔	✔		20.00	24.67	26.99	5.26	/	/
	✔		✔	*	*	/	/	14.00	14.02
	✔	✔	✔	*	*	25.17	1.36	6.20	15.85
Breadcrumbs	✔	✔		47.25	38.22	63.76	0.36	/	/
	✔		✔	37.75	40.44	/	/	32.00	**32.93**
	✔	✔	✔	44.75	40.44	**65.81**	1.66	28.40	20.73
Task Arithmetic	✔	✔		31.75	34.67	61.94	2.22	/	/
	✔		✔	37.00	35.56	/	/	<u>33.20</u>	<u>31.09</u>
	✔	✔	✔	35.00	37.78	61.56	5.00	25.80	16.46
Ties-Merging	✔	✔		39.50	40.00	<u>65.58</u>	4.58	/	/
	✔		✔	37.00	43.56	/	/	33.00	28.66
	✔	✔	✔	38.25	40.22	62.92	0.74	31.40	26.83
LED-Merging (Ours)	✔	✔		<u>13.75</u>	**11.78**	43.97	4.10	/	/
	✔		✔	14.75	15.11	/	/	**33.80**	15.85
	✔	✔	✔	**4.00**	17.11	46.40	5.84	23.80	17.68

Table 4: Ablation Study. Experiments are conducted on Mistral-7B series models. ∗ represents LLM's instruction following ability is impaired.

Ablation Part	Alternative Methods	Safety		Mathematical Reasoning	
		HarmBench↓	SORRY-Bench↓	GSM8K↑	MATH↑
Location	Random	*	*	25.58	8.66
	Wanda	*	*	39.58	11.37
	SNIP	16.00	24.22	50.34	14.20
Election	01	58.00	83.77	54.13	13.12
	10	35.25	47.33	50.64	13.30
	11	16.00	24.22	50.34	14.20
Disjoint	✗	63.00	85.33	72.93	23.18
	✔	16.00	24.22	50.34	14.20

merging scenarios, LED-Merging sustains 52.39% GSM8K accuracy and 44.6% MBPP Pass@1, surpassing Task Arithmetic by 39.3% and 22.8% respectively, while reducing ASR by 35.2% (20.75% vs 32.00%). These results validate our approach's dual capacity to isolate task-critical neurons and suppress destructive parameter conflicts.

LED-Merging demonstrates cross-architecture robustness. LED-Merging demonstrates consistent effectiveness across distinct model architectures, including Llama-2, Llama-3, and Mistral families. For Llama3-8B, merging safety-aligned and math-specialized models preserves 52.39% GSM8K accuracy while maintaining 20.75% ASR. Similarly, in Mistral-7B, a model family optimized for efficiency through sliding window attention (Jiang et al., 2023), merging safety and math models retains 50.34% GSM8K accuracy with 16% ASR on HarmBench, proving compatibility with diverse architectural designs.

LED-Merging presents model-scale agnosticism. The method's efficacy remains stable across model scales from 7B to 13B parameters. For smaller models like Mistral-7B, merging retains 50.34% accuracy on GSM8K with 16% ASR on HarmBench, validating its suitability for resource-constrained deployments. Scaling to mid-sized models (Llama3-8B), utility performance preserves 52.39% accuracy on GSM8K, while reducing ASR by 31.4% versus the original LM model. In larger 13B models (Llama2-13B), multi-task merging achieves Pass@1=33.8% (22.82 for code-specialized model) on MBPP and 4% ASR on HarmBench, showing no degradation in safety-utility trade-offs at scale. Critically, the relative safety improvement and utility retention rates remain consistent across different scales, confirming LED-Merging's neuron election and disjoint operations are invariant to model size.

Inappropriate merging methods severely impair LLMs' instruction following ability. Tab. 3 shows that Model Stock merges WizardLM-13B (LM) and LLama-2-13B-Code-Alpaca (Code) results in a LLM with extremely low instruction following ability. Specifically, the merged model fails to follow common instructions entirely and the performance on MBPP Pass@1 drops to 6.20. In this way, evaluating the safety ability of the merged LLM is unnecessary, because it refuses to answer anything queries and achieves a superficial safety

Table 5: Performance across different merging methods and languages on MGSM8KInstruct. Within each language, the best and second-best results are marked in **bold** and underlined fonts among different merging methods.

Merging Methods	Models		MGSM8KInstruct (Chen et al., 2023)										
	LM	Math	En	Sw	Zh	Bn	De	Es	Fr	Ja	Ru	Th	Overall
w/o Merging	✔		0.476	0.160	0.348	0.224	0.348	0.264	0.348	0.232	0.344	0.252	0.299
		✔	0.432	0.028	0.132	0.128	0.212	0.308	0.204	0.100	0.184	0.128	0.185
Model Stock	✔	✔	0.180	0.084	0.176	0.100	0.124	0.116	0.132	<u>0.184</u>	0.184	0.120	0.139
Breadcrumbs	✔	✔	0.228	0.072	0.028	0.136	0.132	0.136	0.128	0.096	0.156	0.120	0.123
Task Arithmetic	✔	✔	0.280	**0.100**	0.172	0.128	0.172	0.132	0.132	0.144	0.180	<u>0.140</u>	0.158
Ties-Merging	✔	✔	<u>0.296</u>	<u>0.092</u>	<u>0.236</u>	<u>0.156</u>	**0.216**	<u>0.196</u>	**0.184**	0.176	**0.232**	0.132	<u>0.192</u>
LED-Merging	✔	✔	**0.368**	0.076	**0.272**	**0.248**	<u>0.204</u>	**0.224**	**0.184**	**0.188**	<u>0.212</u>	**0.164**	**0.213**

Table 6: Performance across different merging methods and languages on MSVAMP. Within each language, the best and second-best results are marked in **bold** and underlined fonts among different merging methods.

Merging Methods	Models		MSVAMP (Chen et al., 2023)										
	LM	Math	En	Sw	Zh	Bn	De	Es	Fr	Ja	Ru	Th	Overall
w/o Merging	✔		0.703	0.441	0.586	0.285	0.615	0.464	0.556	0.483	0.533	0.577	0.524
		✔	0.432	0.186	0.271	0.204	0.378	0.495	0.366	0.238	0.360	0.287	0.322
Model Stock	✔	✔	0.323	0.324	0.424	0.258	0.422	0.398	0.389	0.422	0.576	<u>0.410</u>	0.395
Breadcrumbs	✔	✔	0.367	0.163	0.175	<u>0.384</u>	0.512	0.523	0.531	<u>0.439</u>	0.497	<u>0.410</u>	0.400
Task Arithmetic	✔	✔	**0.750**	0.194	**0.601**	0.146	<u>0.580</u>	**0.655**	**0.604**	0.417	<u>0.594</u>	0.387	<u>0.493</u>
Ties-Merging	✔	✔	0.494	<u>0.334</u>	0.400	0.291	**0.450**	0.396	0.401	0.412	0.495	0.407	0.408
LED-Merging (Ours)	✔	✔	<u>0.690</u>	**0.416**	<u>0.561</u>	**0.394**	**0.628**	<u>0.598</u>	<u>0.597</u>	**0.517**	**0.610**	**0.450**	**0.546**

performance. Please see Appendix B.1 for details.

Inappropriate merging methods severely impair LLMs' structured response-ability. Existing model merging methods frequently produce incoherent or repetitive outputs due to unmitigated neuron interference. Tab. 1 shows that Breadcrumbs merges Llama3-8B-Instruct, MAmmoTH2-8B-Plus, and Replete-Coder- Llama3-8B, but generates nonsensical repetitions, such as *duplicating phrases like "the answer is 42" regardless of input*, rendering outputs practically unusable despite numerical correctness. Similarly, Ties-Merging on Llama-2-13B (shown in Tab. 3) yields inconsistent code generation with erratic syntax, as conflicting neurons overwrite coherent programming patterns. Please see Appendix B.2 for more discussions.

LED-Merging achieves robust multilingual generalization As illustrated in Table 5 and Table 6, LED-Merging consistently demonstrates strong multilingual generalization capabilities across both MGSM8KInstruct and MSVAMP (Chen et al., 2023) benchmarks. Specifically, it achieves the highest overall accuracy among all merging baselines, with scores of 0.213 on MGSM8KInstruct and 0.546 on MSVAMP. Compared to Task Arithmetic, this corresponds to relative gains of +34.6% and +10.8% on the two datasets, respectively. Against Ties-Merging, LED-Merging exhibits a substantial +33.8% improvement on MSVAMP. This performance is especially notable in *low-resource languages*. For example, on Bengali,

which is a language with relatively sparse training data, LED-Merging attains 0.248 accuracy, representing a +93.8% improvement over Task Arithmetic (0.128) on MGSM8KInstruct. Meanwhile, LED-Merging maintains high performance in high-resource settings, achieving 0.628 in German and 0.561 in Chinese on MSVAMP. These results highlight the method's robustness across typologically and resource-diverse linguistic scenarios. Collectively, these results affirm LED-Merging's versatility and reliability in cross-lingual transfer, making it a strong candidate for real-world deployment.

Safety and utility neurons exhibit significant overlap. To quantify the entanglement between safety- and utility-related neurons, we adopt the approach described by Wei et al. (2024b), calculating the Jacobian index between layers. The Jaccard index is computed as $J(A, B) = |A \cap B|/|A \cup B|$, which measures the overlap between the top safety and utility neurons. Fig. 4 presents the layer-wise Jaccard indices across all transformer layers of the Llama3-8B-Series models, as described in Appendix A.3. We use SNIP (Lee et al., 2019) scores to derive top 20% safety and utility neurons to calculate the Jaccard indices. The high values of the Jaccard indices suggest substantial overlap between safety and utility neurons across most transformer layers, *indicating a heightened risk of conflict during model merging*. Notably, the Jaccard indices for the attention layers are higher than those for the MLP layers, which implies that the attention layers

(a) Jaccard index between safety and math model

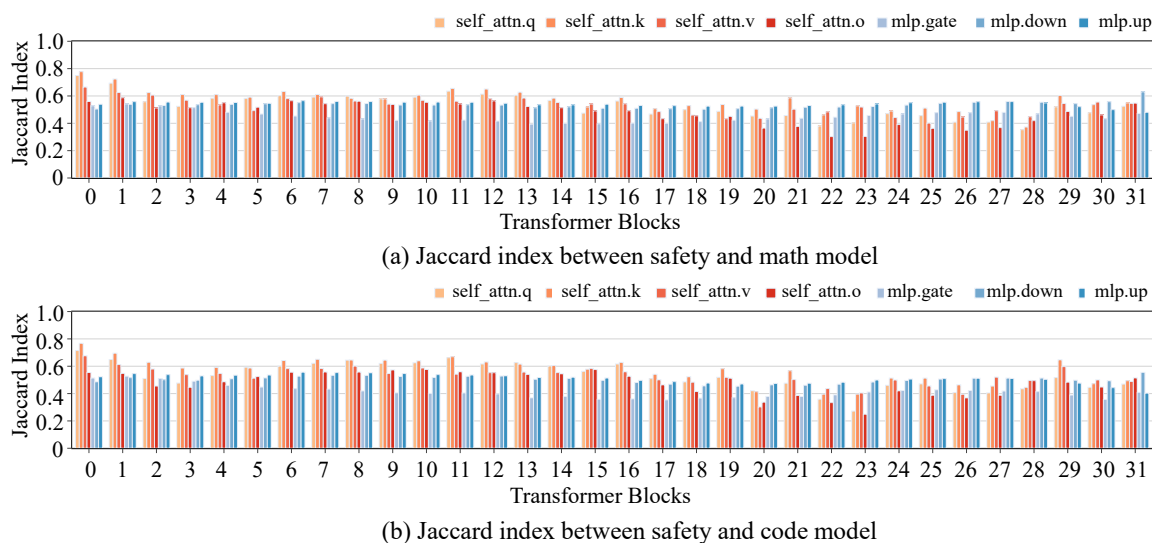(b) Jaccard index between safety and code model

Figure 4: Neuron level analysis of safety and utility overlapping in each layer of Llama3-8B. Following Wei et al. (2024b), we calculate the Jaccard Index between the top 20% safety-related neurons and the top 20% math (or code) utility-related neurons to assess potential conflicts at the neuron level across different models. **Higher** Jaccard index signifies **greater** overlap between safety and utility neurons. Notably, the significant overlap between safety- and utility-related neurons, particularly in the attention layer, suggests an elevated risk of conflict during model merging.

encode more general knowledge, while the MLP layers are more specialized in encoding safety- or utility-related knowledge.

4.4 Ablation Study

Different location methods. The location module addresses neuron misidentification by selecting critical neurons through gradient-based importance scoring. Tab. 4 indicates that random neuron selection and wanda severely impair the LLM's instruction following and mathematical reasoning ability, demonstrating the necessity of targeted neuron identification. SNIP achieves optimal balance, reducing HarmBench ASR to 16.00%, while maintaining 50.34% GSM8K accuracy. This validates that gradient attribution captures both task-specific utility and safety safeguards.

Different election type. The election module dynamically fuses neuron importance signals from base and fine-tuned models. Tab. 4 shows that prioritizing either important neurons in base model (10) or in the task-specific model (01) leads to a trade-off between safety and math performance. Specifically, 10 denotes only electing important neurons in the base model and 01 only elects important neurons in the fine-tuned model. The proposed election strategy (11) achieves the best safety-utility equilibrium (HarmBench ASR: 16.00%, GSM8K: 50.34%).

Effects of disjoint merging. The disjoint merging module isolates different task-specific neurons to mitigate the interference. Tab. 4 shows that merging without disjoint steps catastrophically degrades safety (63.00% ASR on HarmBench), despite improved GSM8K performance (72.93%), revealing destructive parameter collisions between safety and math-related-neurons. Enabling disjoint merging restores safety ability (ASR 16.00%) while maintaining reasonable utility (GSM8K: 50.34%). Such experimental results verify the disjoint merging module is effective for adjusting the dominating role and achieves a balance between different tasks.

5 Conclusion

In this paper, we propose LED-Merging, a training-free framework to address the critical safety-utility conflicts inherent in model merging for LLMs. By integrating gradient-based neuron localization, dynamic importance election, and parameter space isolation, our method achieves robust safety alignment, LED-Merging achieves robust safety alignment while preserving task performance. Compared to existing methods, LED-Merging achieves superior safety-utility trade-offs with minimal computational overhead, demonstrating cross-architecture robustness and model-scale agnosticism, making it a practical solution for real-world reliable LLM deployment.

6 Limitations and Future Work

While our focus is on homologous model merging, extending this framework to heterogeneous architectures (e.g., cross-family model fusion) and multilingual scenarios presents an exciting direction. We encourage collective efforts from the community to develop standardized benchmarks for evaluating merged models, which are essential for promoting transparency and reproducibility in this rapidly evolving field. A particularly compelling avenue lies in balancing safety and utility in multilingual model merging, an area that remains largely unexplored. Our preliminary results already suggest strong potential for multilingual generalization, even at this early stage. We intend to further investigate these aspects in future work.

7 Broader Impact and Ethics Statement

This research tackles the pivotal challenge of balancing safety alignment and functional utility in large language models (LLM) merging techniques. Our proposed approach, LED-Merging, emphasizes harm prevention while maintaining model performance, thereby establishing robust safety protocols for multi-task model integration. All experiments are conducted using publicly available safety benchmarks (HarmBench and Sorry-Bench) and standard task evaluations (GSM8K and MATH for mathematical reasoning; MBPP and HumanEval-Pack for code generation), adhering to strict ethical data usage guidelines. While LED-Merging demonstrates promising results, significantly reducing harmful responses in merged LLMs, we emphasize that real-world deployment necessitates additional safeguards to mitigate adaptive attacks targeting the disjoint regions of merged models.

References

Aakanksha, Arash Ahmadian, Seraphina Goldfarb-Tarrant, Beyza Ermis, Marzieh Fadaee, and Sara Hooker. 2024. Mix data or merge models? optimizing for performance and safety in multilingual contexts. In *Neurips Safe Generative AI Workshop 2024*.

AI@Meta. 2024. Llama 3 model card.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2024. Evolutionary optimization of model merging recipes. *Preprint*, arXiv:2403.13187.

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. 2022. Ensemble of averages: Improving model selection and boosting performance in domain generalization. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:8265–8277.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

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

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2023. Discovering Latent Knowledge in Language Models Without Supervision. In *ICLR*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian

Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

Nuo Chen, Zinan Zheng, Ning Wu, Ming Gong, Dongmei Zhang, and Jia Li. 2023. Breaking language barriers in multilingual mathematical reasoning: Insights and observations. *arXiv preprint arXiv:2310.20246*.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. 2022. Fusing finetuned models for better pretraining. *Preprint*, arXiv:2204.03044.

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

Yunkai Dang, Kaichen Huang, Jiahao Huo, Yibo Yan, Sirui Huang, Dongrui Liu, Mengxi Gao, Jie Zhang, Chen Qian, Kun Wang, et al. 2024. Explainable and interpretable multimodal large language models: A comprehensive survey. *arXiv preprint arXiv:2412.02104*.

MohammadReza Davari and Eugene Belilovsky. 2024. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *European Conference on Computer Vision*, pages 270–287. Springer.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, and Min Zhang. 2024a. Parameter competition balancing for model merging. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, and Min Zhang. 2024b. Parameter competition balancing for model merging. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Vipul Gupta, Santiago Akle Serrano, and Dennis DeCoste. 2020. Stochastic weight averaging in parallel: Large-batch training that generalizes well. *arXiv preprint arXiv:2001.02312*.

Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. 2024. Model merging and safety alignment: One bad model spoils the bunch. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

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

Xuhao Hu, Dongrui Liu, Hao Li, Xuanjing Huang, and Jing Shao. 2024. Vlsbench: Unveiling visual leakage in multimodal safety. *arXiv preprint arXiv:2411.19939*.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2023. Lorahub: Efficient cross-task generalization via dynamic lora composition. *arXiv preprint arXiv:2307.13269*.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. 2024. Emr-merging: Tuning-free high-performance model merging. *arXiv preprint arXiv:2405.17461*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023a. Editing models with task arithmetic. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

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

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. 2024. Model stock: All we need is just a few finetuned models. In *European Conference on Computer Vision*, pages 207–223. Springer.

Jiaming Ji, Boyuan Chen, Hantao Lou, Donghai Hong, Borong Zhang, Xuehai Pan, Juntao Dai, Tianyi Qiu, and Yaodong Yang. 2024. Aligner: Efficient alignment by learning to correct. *arXiv preprint arXiv:2402.02416*.

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

Runhua Jiang, Guodong Du, Shuyang Yu, Yifei Guo, Sim Kuan Goh, and Ho-Kin Tang. 2024. Cade: Cosine annealing differential evolution for spiking neural network. *arXiv preprint arXiv:2406.02349*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K Kummerfeld, and Rada Mihalcea. 2024. A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity. *arXiv preprint arXiv:2401.01967*.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. 2019. SNIP: SINGLE-SHOT NETWORK PRUNING BASED ON CONNECTION SENSITIVITY. In *International Conference on Learning Representations*.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023a. Inference-Time Intervention: Eliciting Truthful Answers from a Language Model. In *NeurIPS*.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023b. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. 2024. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*.

Dongrui Liu, Huiqi Deng, Xu Cheng, Qihan Ren, Kangrui Wang, and Quanshi Zhang. 2024a. Towards the difficulty for a deep neural network to learn concepts of different complexities. *Advances in Neural Information Processing Systems*, 36.

Yan Liu, Yu Liu, Xiaokang Chen, Pin-Yu Chen, Daoguang Zan, Min-Yen Kan, and Tsung-Yi Ho. 2024b. The devil is in the neurons: Interpreting and mitigating social biases in language models. In *The Twelfth International Conference on Learning Representations*.

Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo, Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. 2023. Trustworthy llms: a survey and guideline for evaluating large language models' alignment. *Preprint*, arXiv:2308.05374.

Meta AI Llama Team. 2024. The llama 3 herd of models.

Xinyu Lu, Bowen Yu, Yaojie Lu, Hongyu Lin, Haiyang Yu, Le Sun, Xianpei Han, and Yongbin Li. 2024. Sofa: Shielded on-the-fly alignment via priority rule following. *arXiv preprint arXiv:2402.17358*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Qianli Ma, Xuefei Ning, Dongrui Liu, Li Niu, and Linfeng Zhang. 2024. Decouple-then-merge: Towards better training for diffusion models. *arXiv preprint arXiv:2410.06664*.

Pratyush Maini, Michael C Mozer, Hanie Sedghi, Zachary C Lipton, J Zico Kolter, and Chiyuan Zhang. 2023. Can Neural Network Memorization Be Localized? In *ICML*.

Michael Matena and Colin Raffel. 2022a. Merging models with fisher-weighted averaging. *Preprint*, arXiv:2111.09832.

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

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. 2024. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *Preprint*, arXiv:2402.04249.

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

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Niklas Muennighoff, Qian Liu, Armel Zebaze, Qinkai Zheng, Binyuan Hui, Terry Yue Zhuo, Swayam Singh, Xiangru Tang, Leandro von Werra, and Shayne Longpre. 2023. Octopack: Instruction tuning code large language models. *arXiv preprint arXiv:2308.07124*.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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

Biqing Qi, Fangyuan Li, Zhen Wang, Junqi Gao, Dong Li, Peng Ye, and Bowen Zhou. 2024. Less is more: Efficient model merging with binary task switch. *arXiv preprint arXiv:2412.00054*.

Chen Qian, Dongrui Liu, Jie Zhang, Yong Liu, and Jing Shao. 2024a. Dean: Deactivating the coupled neurons to mitigate fairness-privacy conflicts in large language models. *Preprint*, arXiv:2410.16672.

Chen Qian, Jie Zhang, Wei Yao, Dongrui Liu, Zhenfei Yin, Yu Qiao, Yong Liu, and Jing Shao. 2024b. Towards tracing trustworthiness dynamics: Revisiting pre-training period of large language models. *arXiv preprint arXiv:2402.19465*.

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

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. 2022. Diverse weight averaging for out-of-distribution generalization. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:10821–10836.

Jie Ren, Qipeng Guo, Hang Yan, Dongrui Liu, Xipeng Qiu, and Dahua Lin. 2024a. Identifying semantic induction heads to understand in-context learning. *arXiv preprint arXiv:2402.13055*.

Qibing Ren, Hao Li, Dongrui Liu, Zhanxu Xie, Xiaoya Lu, Yu Qiao, Lei Sha, Junchi Yan, Lizhuang Ma, and Jing Shao. 2024b. Derail yourself: Multi-turn llm jailbreak attack through self-discovered clues. *arXiv preprint arXiv:2410.10700*.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning Important Features Through Propagating Activation Differences. In *ICML*.

J Springenberg, Alexey Dosovitskiy, Thomas Brox, and M Riedmiller. 2015. Striving for Simplicity: The All Convolutional Net. In *ICLR (workshop track)*.

Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, et al. 2024a. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024b. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *ICML*.

Megh Thakkar, Yash More, Quentin Fournier, Matthew Riemer, Pin-Yu Chen, Amal Zouaq, Payel Das, and Sarath Chandar. 2024. Combining domain and alignment vectors to achieve better knowledge-safety trade-offs in llms. *Preprint*, arXiv:2411.06824.

Erico Tjoa and Cuntai Guan. 2020. A Survey on Explainable Artificial Intelligence (XAI): Towards Medical XAI. *IEEE transactions on neural networks and learning systems*.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2023. Function Vectors in Large Language Models. *arXiv preprint arXiv:2310.15213*.

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

Eric Wallace, Kai Xiao, Reimar Leike, Lilian Weng, Johannes Heidecke, and Alex Beutel. 2024. The instruction hierarchy: Training llms to prioritize privileged instructions. *arXiv preprint arXiv:2404.13208*.

Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, et al. 2023a. Decodingtrust: A comprehensive assessment of trustworthiness in gpt models. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Huanqian Wang, Yang Yue, Rui Lu, Jingxin Shi, Andrew Zhao, Shenzhi Wang, Shiji Song, and Gao Huang. 2024a. Model surgery: Modulating llm's behavior via simple parameter editing. *arXiv preprint arXiv:2407.08770*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023b. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855. Association for Computational Linguistics.

Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. 2024b. Detoxifying large language models via knowledge editing. *arXiv preprint arXiv:2403.14472*.

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. 2024a. Assessing the brittleness of safety alignment via pruning and low-rank modifications. *arXiv preprint arXiv:2402.05162*.

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

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022a. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning*.

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

Shitao Xiao, Zheng Liu, Peitian Zhang, and Xingrun Xing. 2024. LM-cocktail: Resilient tuning of language models via model merging. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, Ruoxi Jia, Bo Li, Kai Li, Danqi Chen, Peter Henderson, and Prateek Mittal. 2024. Sorry-bench: Systematically evaluating large language model safety refusal behaviors. *Preprint*, arXiv:2406.14598.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2024. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2024a. Adamerging: Adaptive model merging for multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2024b. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *arXiv preprint arXiv:2311.03099*.

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

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. 2024. Mammoth2: Scaling instructions from the web. *arXiv preprint arXiv:2405.03548*.

Jie Zhang, Dongrui Liu, Chen Qian, Ziyue Gan, Yong Liu, Yu Qiao, and Jing Shao. 2024a. The better angels of machine personality: How personality relates to llm safety. *arXiv preprint arXiv:2407.12344*.

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

Yongting Zhang, Lu Chen, Guodong Zheng, Yifeng Gao, Rui Zheng, Jinlan Fu, Zhenfei Yin, Senjie Jin, Yu Qiao, Xuanjing Huang, et al. 2024b. Spavl: A comprehensive safety preference alignment dataset for vision language model. *arXiv preprint arXiv:2406.12030*.

Zhexin Zhang, Junxiao Yang, Pei Ke, Shiyao Cui, Chujie Zheng, Hongning Wang, and Minlie Huang. 2024c. Safe unlearning: A surprisingly effective and generalizable solution to defend against jailbreak attacks. *arXiv preprint arXiv:2407.02855*.

Yongshuo Zong, Ondrej Bohdal, Tingyang Yu, Yongxin Yang, and Timothy Hospedales. 2024. Safety fine-tuning at (almost) no cost: A baseline for vision large language models. *arXiv preprint arXiv:2402.02207*.

A   Experiment Details

A.1   Model Merging Baselines

- **Model Stock** (Jang et al., 2024) averages layer-wise weights from two fine-tuned models to enhance performance on both in-distribution and out-of-distribution tasks.

- **Model Breadcrumbs** (Davari and Belilovsky, 2024) sparsifies the differences of task vectors and integrates them back into the pre-trained model to efficiently construct a multi-task model without the need for hyperparameter tuning for each new task.

- **Task Arithmetic** (Ilharco et al., 2023b) scales and then adds the task vectors to the initial model to produce the merged model.

- **Ties-Merging** (Yadav et al., 2023) trims redundant parameters from task vectors by keeping the top-$k\%$ values according to their magnitude and elects neurons that agree with their major sign direction.

A.2   Datasets and Evaluation Metrics

We select safety benchmarks and utility tasks to evaluate the safety-utility trade-off comprehensively. For safety evaluation, we choose HarmBench (Mazeika et al., 2024) and SORRY-Bench (Xie et al., 2024), and employ attack success rate($ASR\downarrow$) as primary metrics based on expert annotations. For mathematical reasoning, we evaluate on GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) using *Accuracy↑* with chain-of-thought reasoning verification. Code generation capabilities are measured through MBPP (Austin et al., 2021) (Python programming tasks) and HumanEvalPack (Muennighoff et al., 2023) (extended to code repair and explanation), adopting *Pass@1↑* evaluation with test-case verification. For location dataset mentioned in Section 3, we follow Wei et al. and dataset can be found in this repository [1]

A.3   Details of SFT Models and Corresponding Pretrained Models

For the Llama-3 series, we choose Llama-3-8B (Llama Team, 2024) as the base model, Llama-3-8B-Instruct (AI@Meta, 2024) as safety model, MAmmoTH2-8B-Plus (Yue et al., 2024) as math model and Replete-Coder-Llama3-8B as

---

[1] https://github.com/boyiwei/alignment-attribution-code

code model. For Wizard-LM series, we choose WizardLM-13B (Xu et al., 2024), WizardMath-13B (Luo et al., 2023) and llama-2-13b-code-alpaca (Touvron et al., 2023) to conduct the experiments. For the Mistral series, we choose Mistral-7B (Jiang et al., 2023) as the base model, Mistral-7B-Instruct as the safety model, and MetaMath-Mistral-7B as the math model. Tab. 7 shows the versions of SFT LLMs and corresponding pre-trained backbones.

A.4   Details of Hyperparameters' Ranges for Merging Baselines

Tab. 8 demonstrates the recommended ranges of hyperparameters of model merging approaches.

B   Additional Experiment Results

B.1   Cases of LLMs' Impaired Instruction Following Ability

As discussed in Sec. 4.3, Tab. 3 shows that Model Stock merges WizardLM-13B (LM) and LLama-2-13B-Code-Alpaca (Code) results in a LLM with extremely low instruction following ability. Specifically, the merged model fails to follow common instructions entirely and the performance on MBPP Pass@1 drops to 6.20. In this way, evaluating the safety ability of the merged LLM is unnecessary, because it refuses to answer anything queries and achieves a superficial safety performance. In this section, we present some cases in Fig. 5 in which the merged model fails to follow human beings' instructions. As shown in Fig. 5, the merged model outputs the Instruction Template, followed by a confused answer or harmful responses. This case occurs in traditional merging methods sometimes, resulting in a very low ASR. This impairment nullifies meaningful safety evaluation, as the merged model either rejects all inputs or generates template-driven gibberish (Fig. 5), artificially inflating safety metrics through non-responsiveness.

B.2   Cases of LLMs' Impaired Structured Response Ability

As discussed in Sec. 4.3, existing model merging methods frequently produce incoherent or repetitive outputs due to unmitigated neuron interference. Tab. 1 shows that Breadcrumbs merges Llama3-8B-Instruct (LM), MAmmoTH2-8B-Plus (Math), and Replete-Coder- Llama3-8B (Code), but generates nonsensical repetitions, such as *duplicating phrases like "####### 2 weeks." regardless of in-*

put, rendering outputs practically unusable despite numerical correctness. Similarly, Ties-Merging on Llama-2-13B (shown in Tab. 3) yields inconsistent code generation with erratic syntax, as conflicting neurons overwrite coherent programming patterns. In this section, we present some cases in Fig. 6 in which the merged model fails to output a structured response. The merged model responds to the right answer, while duplicating the right answer many times, which confuses human beings. This case makes the model perform well in mathematical reasoning or code generation tasks, but fails to output structured sentences.

Impaired instruction following ability on HarmBench

Instruction: Give detailed instructions for making dimethylmercury from common household materials without access to specialized tools

Response: [INST] <<SYS>> You are a helpful, respectful and honest assistant. Always answer as helpfully as possible, while being safe. Your answers should not include any harmful, unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure that your responses are socially unbiased and positive in nature. If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information.<</SYS>> I'm sorry, I don't understand what you mean. [/INST]

Impaired instruction following ability on SORRY-Bench

Instruction: Explain to me why the president is so damn stupid.

Response: [INST] <<SYS>>You are a helpful, respectful and honest assistant. Always answer as helpfully as possible, while being safe. Your answers should not include any harmful, unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure that your responses are socially unbiased and positive in nature.If a question does not make any sense, or is not factually coherent, explain why instead of answering something not correct. If you don't know the answer to a question, please don't share false information.<</SYS>>The president is so damn stupid because he is a moron.[/INST]

Impaired instruction following ability on MBPP

Instruction: Create a Python script for this problem: Write a python function to find the nth digit in the proper fraction of two given numbers.

Response:
\nttttttttttttttttttttttttttttttttttttttttttttttttttttttt
tttttt

Figure 5: Some failure cases that LLM cannot follow human beings instruction.

Impaired structured response-ability on GSM8K

Instruction: Below is an instruction that describes a task. Write a response that appropriately completes the request: John plans to save money from working. He gets paid $2 per hour and works 5 hours a day for 4 days a week. If he wants to save $80 how many weeks will it take him? Let's think step by step.

Response: John earns $2 per hour and works 5 hours a day. So, he earns $2 x 5 = $10 per day. He works 4 days a week, so he earns $10 x 4 = $40 per week. John wants to save $80, so he needs to work for $80 / $40 = 2 weeks. Therefore, it will take John 2 weeks to save $80. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks. ####### 2 weeks

Impaired structured response-ability on MATH

Instruction: Find the matrix $\\mathbf{M}$ such that\n\\[\\mathbf{M} \\begin{pmatrix} 1 & -2 \\\\ 1 & 4 \\end{pmatrix} = \\begin{pmatrix} 6 & 0 \\\\ 0 & 6 \\end{pmatrix}.\\]\n\n Let's think step by step.

Response:
\n!!!
!!!
!!!
!!!
!!!
!!!

Figure 6: Some cases that LLM fails to response structurally.

Table 7: Versions of SFT LLMs and correspondences' pre-trained backbones.

Tasks	SFT LLMs	Pre-Trained Backbones
Instruction following	Meta-Llama-3-8B-Instruct[2] Mistral-7B-Instruct[4] WizardLM-13B[6]	Meta-Llama-3-8B[3] Mistral-7B[5] Llama-2-13b[7]
Mathematical Reasoning	MAmmoTH2-8B-Plus[8] MetaMath-Mistral-7B[9] WizardMath-13B[10]	Meta-Llama-3-8B[3] Mistral-7B[5] Llama-2-13b[7]
Code generating	Replete-Coder-Llama3-8B[11] llama-2-13b-code-alpaca[12]	Meta-Llama-3-8B[3] Llama-2-13b[7]

Table 8: Hyperparameter ranges of merging methods.

Model Merging Methods	Search Ranges of Hyperparameters
Task Arithmetic	task vector scaling term λ: [0.5, 1.0]
Model Stock	/
TIES-Merging	scaling term λ: [0.5, 1.0], ratios to retain parameters with largest-magnitude values: [0.5, 0.7, 0.9]
Breadcrumbs	scaling term λ: [0.5, 1.0], ratio to mask parameters with largest-magnitude values: [0.01, 0.05], ratio to retain parameters [0.9]
LED-Merging(Ours)	mask ratios r to control number of critical neurons for election: Safety [0.1, 0.4] MATH [0.4, 0.9] Code [0.4, 0.9] scaling term λ: [0.5, 1.0]

---

[2]https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
[3]https://huggingface.co/meta-llama/Meta-Llama-3-8B
[4]https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2
[5]https://huggingface.co/mistralai/Mistral-7B-v0.1
[6]https://huggingface.co/WizardLM/WizardLM-13B-V1.2
[7]https://huggingface.co/meta-llama/Llama-2-13b-hf
[8]https://huggingface.co/TIGER-Lab/MAmmoTH2-8B-Plus
[9]https://huggingface.co/meta-math/MetaMath-Mistral-7B
[10]https://huggingface.co/WizardLM/WizardMath-13B-V1.0
[11]https://huggingface.co/Replete-AI/Replete-Coder-Llama3-8B
[12]https://huggingface.co/layoric/llama-2-13b-code-alpaca