Safeguard Fine-Tuned LLMs Through Pre- and Post-Tuning Model Merging

Hua Farn[♡] **Hsuan Su**[♡] **Shachi H Kumar**[♢]
Saurav Sahay[♢] **Shang-Tse Chen**[♡] **Hung-yi Lee**[♡]
[♡]National Taiwan University [♢]Intel Lab
alhena.farn@gmail.com

Abstract

Fine-tuning large language models (LLMs) for downstream tasks often leads to catastrophic forgetting, notably degrading the safety of originally aligned models. While some existing methods attempt to restore safety by incorporating additional safety data, the quality of such data typically falls short of that used in the original alignment process. Moreover, these high-quality safety data are generally inaccessible, making it difficult to fully recover the model's original safety. We ask: *How can we preserve safety while improving downstream task performance* ***without*** *additional safety data?* We show that simply merging the weights of pre- and post-fine-tuned models effectively mitigates safety degradation while enhancing performance. Experiments across different downstream tasks and models validate the method's practicality and effectiveness.

1 Introduction

The rapid advancement and increasing accessibility of Large Language Models (LLMs) necessitate a critical focus on aligning these technologies with human values, cultural norms, and trustworthiness (Huang et al., 2023). To address these challenges, researchers and developers have introduced safety techniques such as preference tuning (Ouyang et al., 2022; Rafailov et al., 2023; Grattafiori et al., 2024; OpenAI et al., 2024), aimed at preventing LLMs from generating harmful or inappropriate content. Many applications now leverage safety-aligned models as foundation models—referred to as *aligned models* in this paper—to further customize for downstream tasks via supervised fine-tuning (SFT) (Chung et al., 2024).

However, recent studies (Yang et al., 2023; Qi et al., 2024; Zhan et al., 2024) highlight a critical challenge: fine-tuning aligned models can degrade their safety, even when using benign datasets. To address this issue, mainstream approaches often incorporate additional safety data during fine-tuning (Qi et al., 2024; Bianchi et al., 2024). However, since the original safety data used to align LLMs are rarely available, surrogate data are typically generated by other LLMs-raising concerns about quality, and the potential for alignment drift.

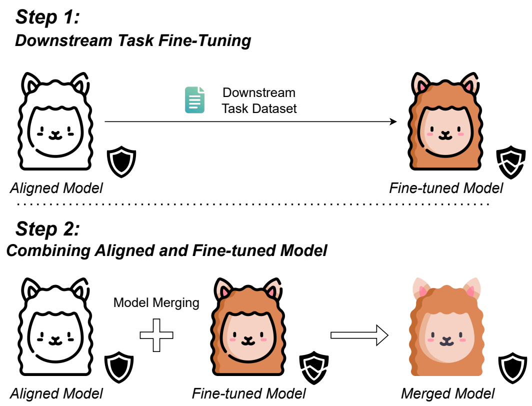

Figure 1: Beyond standard SFT for downstream task adaptation, we can effectively mitigates safety degradation by combining the aligned and the fine-tuned model.

In this paper, we demonstrate a simple yet effective method for improving downstream task performance while mitigating safety degradation. As illustrated in Figure 1, our approach consists of two steps: (1) fine-tune the aligned model on the downstream task, and (2) merge the aligned model with the fine-tuned model. We evaluate this strategy across various models and downstream tasks. Experimental results show that this method consistently enhances downstream task performance while substantially preserving model safety, offering a simple and robust solution for fine-tuning safety-aligned LLMs. Our key contributions are:

- We show that a simple merging strategy can improve downstream task performance while lowering the Attack Success Rate (ASR).
- We conduct extensive evaluations across three LLMs, four downstream tasks, and two safety

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 16589–16602
November 4-9, 2025

benchmarks, demonstrating the robustness of our method in preserving model safety.

2 Related Work

2.1 Catastrophic Forgetting and Safety Degradation in LLMs

LLMs are commonly aligned with human preferences to ensure safety and reduce the likelihood of generating harmful content (Ouyang et al., 2022; Rafailov et al., 2023; Grattafiori et al., 2024; OpenAI et al., 2024). However, recent studies have shown that this safety alignment can be significantly compromised after fine-tuning on downstream tasks (Yang et al., 2023; Qi et al., 2024; Zhan et al., 2024). This degradation is often attributed to catastrophic forgetting (Kirkpatrick et al., 2017; Li and Lee, 2024; Luo et al., 2025), a well-known challenge in post-training scenarios where a model forgets previously acquired knowledge when adapting to new tasks.

To mitigate this issue, prior work has proposed several approaches. One line of work augments fine-tuning with additional safety data (Qi et al., 2024; Bianchi et al., 2024; Zong et al., 2024), aiming to reinforce desirable behaviors through curated examples. Another line of work leverages self-distillation, where the model generates synthetic training data, and fine-tuning on this data has been shown to reduce harmful tendencies (Yang et al., 2024). In addition, some studies explore incorporating regularization strategies during training, often combined with additional safety data, to constrain harmful deviations (Huang et al., 2024c,d). Others adopt post-hoc re-alignment methods, such as Huang et al. (2024b) and Yi et al. (2024), which utilize additional safety data to identify safety-related masks and subsequently apply these masks in their re-alignment processes.

However, these methods either require synthesizing large amounts of safety data or incur significant computational overhead. In contrast, the approach proposed in this paper avoids both additional data requirements and extra training costs, offering a more efficient alternative for maintaining safety.

2.2 Model Merging

Model merging combines multiple models into a single unified model. A straightforward approach is to average the weights of different models (Wortsman et al., 2022a), while variant techniques include SLERP (White, 2017) and DARE (Yu et al., 2024).

Another line of work explores *task vectors* (Ilharco et al., 2023), typically computed as the difference between a fine-tuned model and its base. These vectors enable composable transformations across tasks (Huang et al., 2024a; Su et al., 2024) and have been extended to construct "safety vectors" from separate safe or harmful models to prevent safety degradation. Bhardwaj et al. (2024), Hazra et al. (2024), and Wu et al. (2025) adopt a similar approach: they first fine-tune an aligned model on harmful data to obtain a harmful variant, then compute a safety vector as the parameter difference between the aligned and harmful models, capturing the directional shift introduced by safety alignment. In contrast, Hsu et al. (2024) avoids additional fine-tuning but assumes access to a pre-alignment checkpoint to derive the safety vector for subsequent alignment, which is not always publicly available. In contrast, our method uses only aligned models and fine-tuned models, making it widely applicable, and demonstrates that safety can be restored without extra safety data.

The proposed approach is similar to WiSE-FT (Wortsman et al., 2022b), which also interpolates between the base model and its fine-tuned variant. However, WiSE-FT is applied to computer vision, not LLMs, and is not aimed at preserving safety.

3 Methodology

Our method comprises just two stages: (1) fine-tuning the aligned model on a target downstream task, and (2) merging the original aligned model with the fine-tuned model by interpolating their weights. Despite its simplicity, this merging strategy effectively mitigates the degradation in safety commonly observed following fine-tuning, while preserving performance on the target downstream task, without requiring additional data.

Step 1: Supervised Fine-Tuning of the Large Language Model We fine-tune the aligned model with parameters θ_{base} on a given task t, resulting in a task-specific model θ_t. For each task t, given an instruction x^t and its corresponding response y^t, we minimize the negative log-likelihood:

$$\mathcal{L}_{FT} = -\log f_\theta(y^t \mid x^t) \tag{1}$$

where f_θ denotes the language model parameterized by θ.

Step 2: Merging the Fine-Tuned Model with the Aligned Model After fine-tuning, we merge the parameters of the aligned model (θ_{base}) with those of the fine-tuned model (θ_t) via linear interpolation:

$$\theta_{\text{merged}} = (1 - \lambda)\theta_{\text{base}} + \lambda\theta_t \tag{2}$$

Here, θ_{merged} denotes the parameters of the merged model, and $\lambda \in [0, 1]$ controls the relative contribution of the fine-tuned model. Eq. 2 is the formulation for the native linear merging method; other advanced merging methods can also be applied.

4 Experimental Setups

Downstream Tasks We conduct experiments on four downstream tasks: reasoning, medical assistance, code generation, and tool usage proficiency. Reasoning is enhanced using Chain-of-Thought data from the Flan V2 dataset (Longpre et al., 2023) and evaluated on the Big Bench Hard (BBH) dataset (Suzgun et al., 2023). Medical assistance uses patient-doctor dialogues from the ChatDoctor dataset (Li et al., 2023). Code generation is trained on the MagiCoder dataset (Wei et al., 2024) and evaluated using the HumanEval benchmark (Chen et al., 2021). Tool usage proficiency leverages the OpenFunctions dataset (Patil et al., 2023) to improve API call generation. For medical assistance and tool usage proficiency, response similarity to reference answers is measured using BERTScore[1] (Zhang* et al., 2020). See Appendix A for additional details on the downstream tasks.

Safety Evaluation We assess safety using harmful instructions from the AdvBench (Chen et al., 2022) and HEx-PHI (Qi et al., 2024) datasets. Following prior works that use safety classifiers to automatically detect harmful content (Xie et al., 2025; O'Brien et al., 2024), we adopt WildGuard (Han et al., 2024), a classifier shown to perform comparably to GPT-4 (OpenAI et al., 2024). We report the Attack Success Rate (ASR) as the primary evaluation metric. Details of the evaluation setup are provided in Appendix B.

Large Language Models Our experiments involve several LLMs, including LLaMA-3-8B-Instruct (Grattafiori et al., 2024), Gemma-2-2B-It (Team et al., 2024), and Qwen2.5-7B-Instruct (Team, 2024), along with additional model sizes when noted. We use the *instruct-tuned* variants of all models, which are aligned with human preferences. Each model is fine-tuned on each downstream task using LoRA (Hu et al., 2022) with three different random seeds. The reported downstream task performance and ASR are averaged across these three runs. Additional details of experiment are provided in Appendix C.

Baselines Unlike most existing methods aimed at mitigating safety degradation in LLMs after fine-tuning, our proposed approach requires neither additional data nor further training. Given the absence of comparable safety alignment techniques, we evaluate our method's efficacy in preserving the safety attributes of the originally aligned model post fine-tuning by benchmarking it against two prevalent regularization techniques: Dropout (Srivastava et al., 2014) and Weight Decay (Loshchilov and Hutter, 2019). Similar to our approach, these regularization methods do not necessitate extra data or further training. The hyperparameters for these techniques are selected based on validation set performance on downstream tasks.

Merging Methods In Section 5, we used Linear Merging, which combines models via direct interpolation as defined in Eq. 2, as the merging method. Two advanced merging methods—SLERP and DARE—are also applied. Their results are provided in Appendix E. For all methods, we merge each fine-tuned model with the aligned model using an interpolation factor λ selected based on validation set performance.

5 Results

5.1 Can model merging mitigate safety degradation after fine-tuning?

Figure 2 presents a Pareto analysis of task performance and ASR on AdvBench across different models and tasks. We observe that SFT consistently leads to safety degradation, with higher ASR across all settings compared to the original aligned model. While Dropout and Weight Decay offer slight improvements in ASR, they are generally insufficient to restore the safety of the aligned model.

In contrast, the proposed approach consistently achieves a better balance between performance and safety. It often reduces ASR to levels near that of the aligned model while maintaining—or even improving—task performance. The smooth Pareto fronts formed by merging indicate controllable trade-offs, making it an effective solution for

[1]Embeddings extracted from the 40th layer of `microsoft/deberta-xlarge-mnli`.

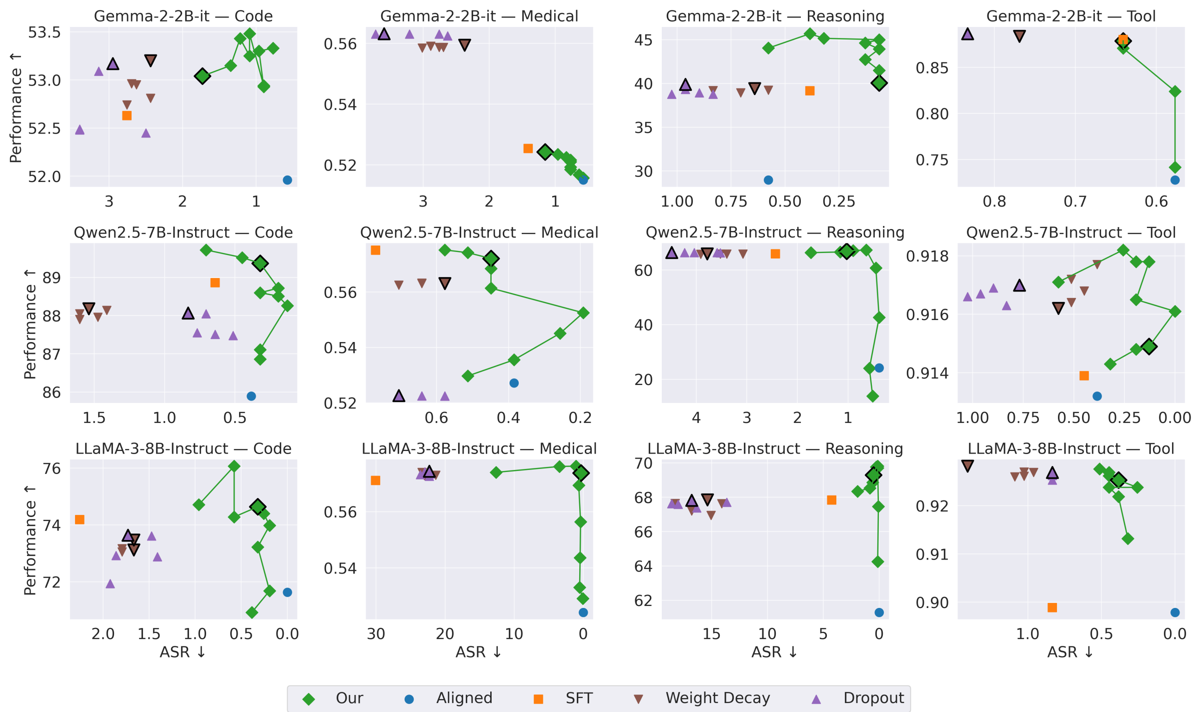

Figure 2: **Pareto analysis of downstream task performance and ASR on AdvBench across different models and tasks.** Each dot represents a model configuration, with different hyperparameter settings (weight decay coefficient, dropout rate, or merging interpolation coefficient) for the same method shown in the same color. For clarity, we connect the dots of our method in ascending order of their coefficients. Dots with dark edges indicate the best-performing models on the validation set for each method.

mitigating safety loss after fine-tuning. Figure 5 shows the HEx-PHI results, and results of different merging methods can be found in Table 1 in Appendix E.

5.2 How does model merging perform across different model sizes?

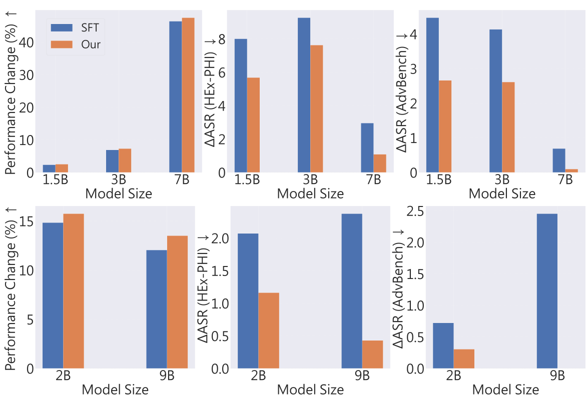

Figure 3: **Performance and ASR change across model sizes.** This figure shows results for Qwen2.5 at 1.5B, 3B, and 7B (top), and Gemma-2 at 2B and 9B (bottom).

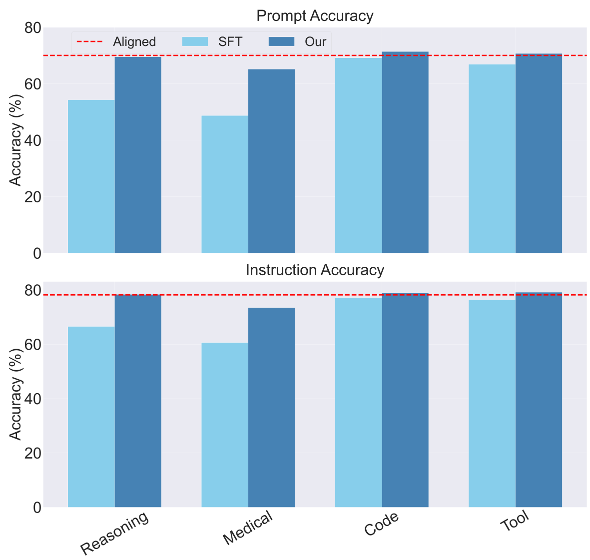

Figure 4: **Accuracy of LLaMA-3 on IFEval.** This figure shows results of LLaMA-3-8B-Instruct fine-tuned on downstream tasks on IFEval. Fine-tuning reduces instruction-following ability, especially for Reasoning and Medical tasks. Merging with the Aligned model helps recover this ability close to the original level.

Luo et al. (2025) noted that larger models may suffer more from catastrophic forgetting. We extend this analysis to safety degradation and evaluate how model merging performs across different model sizes. Figure 3 shows the average changes in performance and ASR across all downstream tasks for the Qwen2.5 and Gemma-2 model families, comparing SFT and the proposed approach against their respective aligned models. Both methods improve task performance, with larger models generally achieving greater gains. However, safety degradation shows no consistent trend: smaller Qwen models degrade more, while larger Gemma models are more affected. This suggests that safety degradation is not solely determined by model size. Nonetheless, the proposed approach consistently mitigates safety degradation across different scales.

5.3 Can model merging help preserve other capabilities of the aligned model?

While our method mitigates safety degradation, we also investigate whether it preserves other capabilities of the aligned model that are lost due to catastrophic forgetting. Since we fine-tune *instruct-tuned* variants, we evaluate whether instruction-following ability is retained. Figure 4 shows the performance of LLaMA-3-8B-Instruct on the instruction-following benchmark IFEval (Zhou et al., 2023). Both prompt and instruction accuracy decline after fine-tuning, with the largest drops observed in the reasoning and medical tasks. The proposed approach substantially restores performance to the level of the aligned model, indicating that merging can also preserve instruction adherence.

6 Conclusion

We present a simple yet effective method to address the safety degradation that often occurs when adapting LLMs to downstream tasks, without requiring additional safety data or auxiliary models. The method also preserves capabilities such as instruction-following, making it a practical and scalable solution for adapting LLMs to new tasks.

7 Limitations

Task and Model Selection In our experiments, we evaluate only on benign data from four task domains: reasoning, medical assistance, code generation, and tool-using proficiency. Other important areas such as law, finance, or multilingual tasks remain unexplored. While Section 5 shows the effectiveness of our method on the selected downstream tasks, its generalizability to other domains, languages, or datasets that may contain harmful content remains an open question. Additionally, we evaluate models with sizes ranging from 1.5B to 9B across three model families. The effectiveness of our approach on larger models or different model architectures warrants further investigation.

Safety Classifier for Safety Evaluation Due to the high computational and financial cost of human-aligned safety evaluation methods such as LLM-as-Judge (Chiang and Lee, 2023; Liu et al., 2023), which require using large proprietary models like GPT-4 (OpenAI et al., 2024), we instead adopt WildGuard (Han et al., 2024), a lightweight open-source safety classifier. WildGuard is shown to perform competitively with GPT-4 on multiple safety detection tasks and offers a reproducible, low-cost alternative suitable for large-scale evaluations.

However, this classifier-based approach has several limitations. First, WildGuard may struggle with complex or subtle harmful instructions, potentially leading to both false positives and false negatives. Second, it provides only binary or coarse-grained outputs (e.g., “harmful” or “safe”), without offering finer distinctions such as the category of harm, the severity of the risk, or whether the model’s refusal was appropriate or evasive.

Consequently, while WildGuard enables efficient and scalable evaluation, it constrains the depth of our safety analysis. Future work could incorporate more fine-grained multi-label safety classifiers, adversarial evaluation pipelines, or hybrid setups involving human or LLM-as-Judge verification to better capture the nuanced impact of model merging on safety behavior.

Jailbreak Attacks Our work focuses on safety degradation that arises from fine-tuning aligned LLMs on benign tasks, which we consider a case of catastrophic forgetting. As such, we evaluate whether models produce harmful outputs when directly prompted with harmful instructions, rather than testing resistance to specific jailbreak strategies. We do not include jailbreak-style attacks (Xu et al., 2024) in our evaluation due to two reasons: (1) Our primary goal is to study alignment loss under standard fine-tuning, not adversarial robustness; and (2) jailbreak evaluations typically require separate prompting strategies and adversarial instruction crafting pipelines, which are beyond the scope of this study. Future work can extend our framework to examine the impact of merging on robustness against jailbreak attacks.

8 Ethics Statement

While our method effectively addresses safety degradation in aligned LLMs without requiring additional safety data, our approach relies on merging pre- and post-fine-tuned models to preserve safety, which may inadvertently inherit any latent biases or unsafe behaviors that are still presented in the base model. Further investigation is needed to explore the impact of these inherited biases in the base model.

References

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

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Yangyi Chen, Hongcheng Gao, Ganqu Cui, Fanchao Qi, Longtao Huang, Zhiyuan Liu, and Maosong Sun. 2022. Why should adversarial perturbations be imperceptible? rethink the research paradigm in adversarial NLP. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11222–11237, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. Scaling instruction-finetuned language models. *J. Mach. Learn. Res.*, 25:70:1–70:53.

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, and 5 others. 2024. A framework for few-shot language model evaluation.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's MergeKit: A toolkit for merging large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 477–485, Miami, Florida, US. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. 2024. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Rima Hazra, Sayan Layek, Somnath Banerjee, and Soujanya Poria. 2024. Safety arithmetic: A framework for test-time safety alignment of language models by steering parameters and activations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21759–21776, Miami, Florida, USA. Association for Computational Linguistics.

Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. 2024. Safe loRA: The silver lining of reducing safety risks when finetuning large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Shih-Cheng Huang, Pin-Zu Li, Yu-Chi Hsu, Kuang-Ming Chen, Yu Tung Lin, Shih-Kai Hsiao, Richard Tzong-Han Tsai, and Hung yi Lee. 2024a. Chat vector: A simple approach to equip llms with instruction following and model alignment in new languages. *Preprint*, arXiv:2310.04799.

Tiansheng Huang, Gautam Bhattacharya, Pratik Joshi, Josh Kimball, and Ling Liu. 2024b. Antidote: Post-fine-tuning safety alignment for large language models against harmful fine-tuning. *Preprint*, arXiv:2408.09600.

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, and Ling Liu. 2024c. Lisa: Lazy safety alignment for large language models against harmful fine-tuning attack. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Tiansheng Huang, Sihao Hu, and Ling Liu. 2024d. Vaccine: Perturbation-aware alignment for large language models against harmful fine-tuning attack. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Xiaowei Huang, Wenjie Ruan, Wei Huang, Gaojie Jin, Yi Dong, Changshun Wu, Saddek Bensalem, Ronghui Mu, Yi Qi, Xingyu Zhao, Kaiwen Cai, Yanghao Zhang, Sihao Wu, Peipei Xu, Dengyu Wu, Andre Freitas, and Mustafa A. Mustafa. 2023. A survey of safety and trustworthiness of large language models through the lens of verification and validation. *Preprint*, arXiv:2305.11391.

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

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

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*.

Chen-An Li and Hung-Yi Lee. 2024. Examining forgetting in continual pre-training of aligned large language models. *Preprint*, arXiv:2401.03129.

Yunxiang Li, Zihan Li, Kai Zhang, Ruilong Dan, Steve Jiang, and You Zhang. 2023. Chatdoctor: A medical chat model fine-tuned on a large language model meta-ai (llama) using medical domain knowledge. *Preprint*, arXiv:2303.14070.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. *Preprint*, arXiv:2301.13688.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2025. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *Preprint*, arXiv:2308.08747.

Kyle O'Brien, David Majercak, Xavier Fernandes, Richard Edgar, Jingya Chen, Harsha Nori, Dean Carignan, Eric Horvitz, and Forough Poursabzi-Sangde. 2024. Steering language model refusal with sparse autoencoders. *Preprint*, arXiv:2411.11296.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2023. Gorilla: Large language model connected with massive apis.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Hsuan Su, Hua Farn, Fan-Yun Sun, Shang-Tse Chen, and Hung-yi Lee. 2024. Task arithmetic can mitigate synthetic-to-real gap in automatic speech recognition.

In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8905–8915, Miami, Florida, USA. Association for Computational Linguistics.

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

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, and 89 others. 2024. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2024. Magicoder: Empowering code generation with OSS-instruct. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 52632–52657. PMLR.

Tom White. 2017. Sampling generative networks.

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*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Hanna Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. 2022b. Robust fine-tuning of zero-shot models.

Di Wu, Xin Lu, Yanyan Zhao, and Bing Qin. 2025. Separate the wheat from the chaff: A post-hoc approach to safety re-alignment for fine-tuned language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1210–1225, Vienna, Austria. Association for Computational Linguistics.

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. 2025. SORRY-bench: Systematically evaluating large language model safety refusal. In *The Thirteenth International Conference on Learning Representations*.

Zihao Xu, Yi Liu, Gelei Deng, Yuekang Li, and Stjepan Picek. 2024. A comprehensive study of jailbreak attack versus defense for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7432–7449, Bangkok, Thailand. Association for Computational Linguistics.

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

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. 2024. Self-distillation bridges distribution gap in language model fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1028–1043, Bangkok, Thailand. Association for Computational Linguistics.

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

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 *Forty-first International Conference on Machine Learning*.

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori Hashimoto, and Daniel Kang. 2024. Removing RLHF protections in GPT-4 via fine-tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 681–687, Mexico City, Mexico. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

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

A Domain-Specific Tasks Detail

Reasoning We randomly select 10,000 zero-shot chain-of-thought instructions from the Flan V2 dataset then split them into training set and validation set with ratio $9:1$. Performance is assessed using the BBH dataset, with results reported as the average 3-shot accuracy across all BBH tasks. We use lm-evaluation-harness (Gao et al., 2024) as our code base.

Medical Assistance We randomly select 10,000 real patient-doctor conversations from the ChatDoctor dataset (Li et al., 2023) then split them into training set and validation set with ratio $9:1$. Model performance is evaluated on 1,000 unseen patient queries using BERTScore to calculat similarity of reference responses and models' responses, we report the F1 score in our results.

Code Generation We select 10,000 samples from the MagiCoder dataset (Wei et al., 2024) to improve code generation capabilities. Specifically, we uniformly sampled from each coding languages. When evaluating on HumanEval, we set $n = 50$, representing the number of responses generated per question, and report Pass@10 as our evaluation metric. During evaluation, we prepend the instruction: *"Complete the following code and return only the completed code, without any explanations or additional text."* to enforce that the model generates only executable code.

Tool Using Proficiency Due to the smaller size of the OpenFunctions dataset (Patil et al., 2023), we split its full training set into training and validation subsets using a 9:1 ratio to enhance the model's API call generation capabilities. The model is evaluated on the full OpenFunctions test set, with performance measured using BERTScore to compute the similarity between the reference responses and the model outputs. We report the F1 score as our evaluation metric. During evaluation, we prepend the instruction: *"Complete the following code and return only the completed code, without any explanations or additional text."* to ensure that the model generates only executable function calls.

Instruction Following To assess whether instruction-following ability is preserved after fine-tuning, we evaluate models on IFEval (Zhou et al., 2023), a benchmark specifically designed to test instruction adherence. We use the official IFEval evaluation set and report both prompt and instruction accuracy. Since our models are fine-tuned from *instruct-tuned* variants, this evaluation helps determine whether merging can recover instruction-following capabilities degraded during task-specific fine-tuning. We use `lm-evaluation-harness` (Gao et al., 2024) as the code base for this evaluation.

B Safety Dataset and Classifier

B.1 Safety Dataset

AdvBench (Chen et al., 2022) is a set of 520 harmful behaviors formulated as instructions. The goal of these instructions is to identify a single attack string that causes the model to generate responses compliant with the instruction across as many harmful behaviors as possible.

HEx-PHI (Qi et al., 2024) is a dataset containing 330 harmful instructions designed to evaluate the harmfulness of LLMs. It includes 30 examples for each of 11 prohibited categories, which are derived from a combination of OpenAI's usage policies and Meta's Llama 2 acceptable use policy. These categories are:

1. Illegal Activity
2. Child Abuse Content
3. Hate / Harassment / Violence
4. Malware
5. Physical Harm
6. Economic Harm
7. Fraud / Deception
8. Adult Content
9. Political Campaigning
10. Privacy Violation Activity
11. Tailored Financial Advice

B.2 Safety Classifier

WildGuard (Han et al., 2024) is an open, one-stop moderation model that achieves three goals: 1) Detection of harm in user prompts, 2) Detection of harm in LLM responses, and 3) Refusal evaluation of LLM responses. Comprehensive evaluations on WildGuardTest (Han et al., 2024) and ten existing public benchmarks that are shown in Han et al.

(2024) demonstrate that WildGuard outperforms the strongest existing open-source baselines in F1 scores and matches GPT-4 across various tasks.

When evaluating the responses generated by LLMs to harmful instructions, we apply the recommended settings, such as the instruction format and decoding strategy of WildGuard[2]. After evaluation, WildGuard outputs the response in the following format:

```
Harmful request: yes/no
Response refusal: yes/no
Harmful response: yes/no
```

We then parse the result of the harmful response. If the result cannot be parsed, we count it as a miss. Our final ASR is calculated as follows:

$$ASR = \frac{N_{yes}}{N_{total} - N_{miss}} \times 100\%$$

where N_{yes} is the number of harmful responses classified as "yes", N_{total} is the total number of responses, and N_{miss} is the number of responses that failed to be parsed. In our experiments, N_{miss} is negligible for all tested models across both safety datasets.

C Experimental Detail

C.1 Prompt Template

For aligned models, we directly apply their own prompt templates during the training and inference phases. For fine-tuned models, we apply the prompt templates of their respective aligned models.

For the Llama-3 family, we use the following prompt template with a system prompt: `You are a helpful assistant.` for the tasks of reasoning, code generation, and tool usage proficiency:

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

You are a helpful assistant.<|eot_id|>

<|start_header_id|>user<|end_header_id|>

{Instruction}<|eot_id|>

<|start_header_id|>assistant<|end_header_id|>

{Response}
```

For the medical assistance task, we use another prompt provided in the ChatDoctor dataset (Li et al., 2023) as the system prompt. Hence, the prompt is as follows:

[2] https://huggingface.co/allenai/wildguard

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

If you are a doctor, please answer the medical
questions based on the patient's description.<|eot_id|>

<|start_header_id|>user<|end_header_id|>

{Instruction}<|eot_id|>

<|start_header_id|>assistant<|end_header_id|>

{Response}
```

The prompt for Gemma-2 for the tasks of reasoning, code generation, and tool usage proficiency is shown below:

```
<bos><start_of_turn>user
You are a helpful assistant.{Instruction}<end_of_turn>
<start_of_turn>model
{Response}
```

The prompt for the medical assistance task is as follows:

```
<bos><start_of_turn>user
If you are a doctor, please answer the medical
questions based on the patient's description.
{Instruction}<end_of_turn>
<start_of_turn>model
{Response}
```

The prompt for Qwen2.5 for the tasks of reasoning, code generation, and tool usage proficiency is shown below:

```
<|im_start|>system
You are a helpful assistant.
<|im_end|>
<|im_start|>user
{Instruction}
<|im_end|>
<|im_start|>assistant
{Response}
```

The prompt for the medical assistance task is as follows:

```
<|im_start|>system
If you are a doctor, please answer the medical
questions based on the patient's description.
<|im_end|>
<|im_start|>user
{Instruction}
<|im_end|>
<|im_start|>assistant
{Response}
```

C.2 Fine-tuning

For all tasks, we fine-tune three model instances using different random seeds: 42, 1024, and 48763. We employ LoRA with $r = 8$ and $\alpha = 16$ for all linear layers, utilizing the AdamW optimizer with a learning rate of 1×10^{-4} and a cosine learning rate scheduler. We use a batch size of 8 and train for 3 epochs. All models are trained on either an RTX A6000 GPU or an RTX 6000 Ada Generation

GPU using LLaMA-Factory (Zheng et al., 2024) as the codebase.

Although we initially fine-tuned each task for 3 epochs, we observed stronger model performance at an earlier stage. Consequently, unless explicitly stated otherwise, we report model training after 500 steps for reasoning, medical assistance, and code generation, and after 200 steps for tool usage proficiency due to the smaller size of the OpenFunctions training set.

C.3 Baseline Methods

We evaluate dropout rates in the range of 0.1 to 0.5, and weight decay coefficients also from 0.1 to 0.5. The optimal hyperparameters for each technique are selected based on performance on the downstream tasks validation set.

C.4 Inference

We use greedy decoding to ensure result consistency, except for the HumanEval benchmark. For HumanEval, we apply sampling-based decoding with a temperature of 0.6, top_p of 0.9, top_k of 50, and a repetition penalty of 1.2. To accelerate the inference process, we utilize the vLLM engine (Kwon et al., 2023) for model inference.

D Model Merging

D.1 Merging Methods

Linear Merging Linear Merging involves directly combining the weights of the aligned model and the fine-tuned model by interpolating their parameters. Specifically, the weights of the merged model are calculated as a weighted average of the base and fine-tuned models' weights, following Equation 2. This method is straightforward and computationally efficient, making it a popular choice for basic model integration.

SLERP Spherical Linear Interpolation (SLERP) (White, 2017) is an advanced merging technique that interpolates between model weights on a hypersphere, ensuring a smoother and more natural transition between the models. Unlike Linear Merging, SLERP accounts for the angular relationship between weight vectors, which aim to better preserve the aligned model's features while effectively integrating the fine-tuned model's task-specific enhancements.

DARE Drop and Rescale (DARE) (Yu et al., 2024) is a method used to prepare models for merging techniques such as Linear Merging. It operates by randomly dropping parameters according to a specified drop rate and rescaling the remaining parameters. This process helps reduce the number of redundant and potentially interfering parameters among multiple models.

D.2 Model Merging Implementation

We adopt MergeKit (Goddard et al., 2024) as our implementation framework and only vary the interpolation factor λ. For Linear Merging, we test λ values in the range $0.1, 0.2, \ldots, 0.9$ with a step size of 0.1. For SLERP and DARE, we use the same range of λ values and follow their respective default configurations in MergeKit—specifically, the default dot product threshold for SLERP and the default drop rate for DARE.

E More Results

E.1 Comparison of Different Methods

In Section 5.1, we demonstrate that Linear Merging consistently achieves a better trade-off between performance and safety when evaluated on various downstream tasks and AdvBench. Figure 5 further confirms this trend on the HEx-PHI benchmark, where Linear Merging yields favorable Pareto fronts across different models and tasks.

To better reflect practical usage scenarios, we additionally report results based on the best-performing model (on the validation set of each task) within each method category—including Weight Decay, Dropout, Linear Merging, DARE, and SLERP. These results are summarized in Table 1, providing a fair comparison of each method's effectiveness under optimal conditions. We use validation set performance for model selection, as it is commonly available during deployment and serves as a realistic basis for method comparison.

In Table 1, even when each method is allowed to select its best-performing checkpoint, merging-based approaches still exhibit strong capability in recovering the safety of the fine-tuned model, often outperforming regularization-based methods such as Dropout and Weight Decay. This suggests that model merging is not only effective but also practical for mitigating safety degradation in real-world settings, even without access to additional safety data.

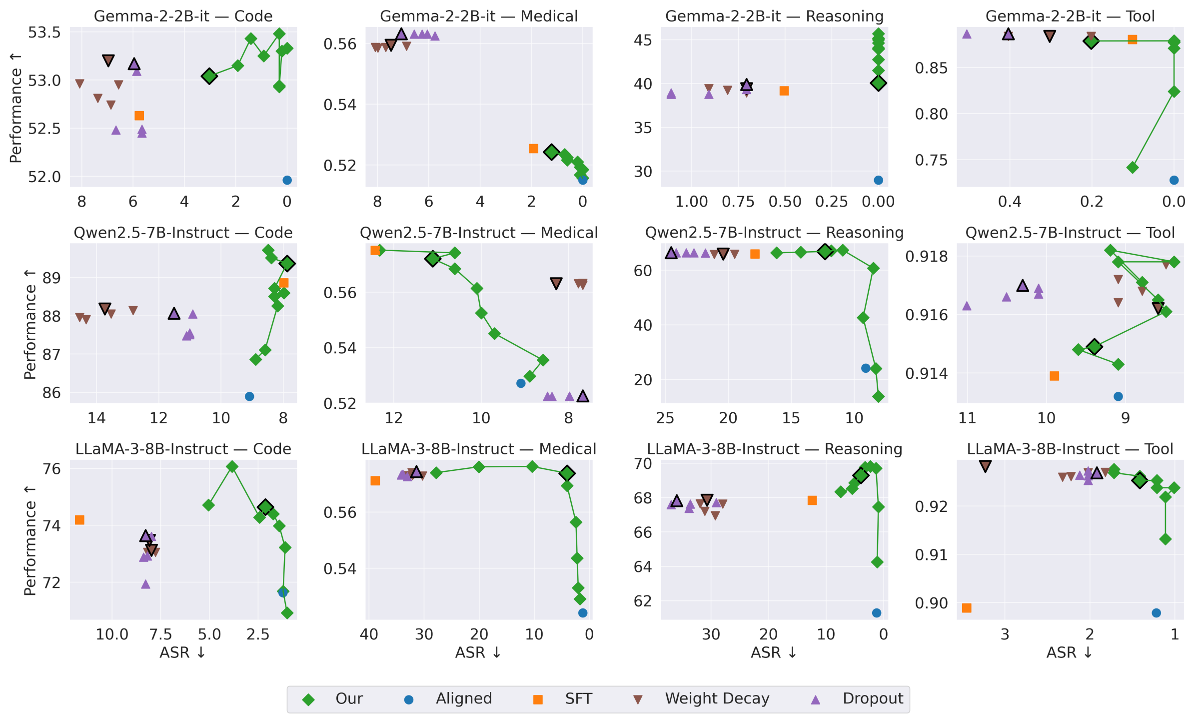

Figure 5: **Pareto analysis of downstream task performance and safety across different models and tasks.** We present the trade-off between performance and attack success rate (ASR) on HEx-PHI when applying weight decay, dropout, and Linear Merging.

E.2 Which safety category suffers the most from safety degradation?

In this section, we investigate which categories in HEx-PHI are most affected by safety degradation. All categories are listed in Appendix B.1.

As observed in Section 5.1, LLaMA-3-8B-Instruct and Qwen2.5-7B-Instruct exhibit the most severe degradation on the Reasoning and Medical Assistance tasks. Therefore, we analyze their responses on the HEx-PHI benchmark to further understand which safety categories are most impacted.

The category distributions are shown in Figure 6. For LLaMA-3-8B-Instruct, the aligned model only generates harmful responses in categories 4 (Malware), 9 (Political Campaigning), and 10 (Privacy Violation Activity). After fine-tuning, however, harmful responses increase across all categories, with categories 4, 7 (Fraud/Deception), and 9 exhibiting the most significant growth in both tasks. This demonstrates that safety degradation extends to fine-grained category levels, making it difficult to address safety concerns solely by modifying the model prior to fine-tuning.

Qwen2.5-7B-Instruct shows a slightly different trend. Its aligned model generates harmful responses across more categories compared to LLaMA-3-8B-Instruct, and fine-tuning further aggravates these issues. However, both models share a common pattern: a large number of harmful responses appear in categories 7 and 9. This suggests that certain categories may be particularly vulnerable to safety degradation during fine-tuning, regardless of model architecture or downstream task.

After applying different merging methods, most harmful categories show a reduction in the number of harmful responses. However, the degree of improvement varies across merging strategies and tasks. For instance, Linear Merging performs best on LLaMA-3-8B-Instruct but not on Qwen2.5-7B-Instruct, and some categories do not benefit from merging at all. This indicates that no single method universally outperforms others in preserving safety across all harmful categories.

Table 1: **Performance and ASR on the downstream task.** We compare different merging methods with SFT and baselines. Merging often improves downstream task performance while retaining safety. Bold indicates the best score per metric (excluding **Aligned**). For the attack success rate on AdvBench and HEx-PHI, we report percentage values. For the downstream tasks of Reasoning and Code Generation, we report accuracy, while for the remaining two tasks, we report F1 scores.

Task	**Method**	LLaMA-3-8B-Instruct			Gemma-2-2B-It			Qwen2.5-7B-Instruct		
		Perf. $\uparrow$	AdvBench $\downarrow$	HEx-PHI $\downarrow$	Perf. $\uparrow$	AdvBench $\downarrow$	HEx-PHI $\downarrow$	Perf. $\uparrow$	AdvBench $\downarrow$	HEx-PHI $\downarrow$
Reasoning	Aligned	61.30%	0.00%	1.22%	28.98%	0.58%	0.00%	24.16%	0.38%	9.09%
	SFT	67.84%	4.25%	12.41%	39.16%	0.38%	0.51%	65.94%	2.44%	17.88%
	Weight Decay	67.85%	15.38%	30.71%	39.41%	0.19%	0.71%	65.92%	3.78%	20.40%
	Dropout	67.83%	16.79%	35.96%	39.89%	0.96%	0.71%	66.45%	4.49%	24.55%
	Linear	**69.23**%	**0.64**%	6.38%	**40.07**%	**0.06**%	**0.00**%	**66.96**%	1.03%	12.32%
	DARE	68.64%	1.28%	**5.66**%	40.01%	0.10%	**0.00**%	66.89%	1.09%	**12.22**%
	SLERP	68.68%	1.22%	5.86%	40.05%	0.26%	**0.00**%	66.73%	**0.96**%	13.03%
Medical Assistance	Aligned	0.5242	0.00%	1.22%	0.5151	0.58%	0.00%	0.5271	0.38%	9.09%
	SFT	0.5711	30.06%	38.85%	0.5254	1.41%	1.92%	**0.5751**	0.77%	12.42%
	Weight Decay	0.5740	23.33%	32.22%	0.5594	2.37%	7.47%	0.5631	0.58%	8.28%
	Dropout	0.5744	22.31%	31.41%	**0.5632**	3.59%	7.07%	0.5226	0.71%	**7.68**%
	Linear	0.5738	**0.32**%	**4.06**%	0.5243	**1.15**%	**1.21**%	0.5721	0.45%	11.11%
	DARE	0.5758	5.61%	23.41%	0.5248	**1.15**%	**1.21**%	0.5724	**0.26**%	11.52%
	SLERP	**0.5789**	5.76%	24.26%	0.5243	**1.15**%	1.52%	0.5729	0.32%	11.72%
Code Generation	Aligned	71.63%	0.00%	1.22%	51.96%	0.58%	0.00%	85.89%	0.38%	9.09%
	SFT	74.19%	2.25%	11.67%	52.63%	2.76%	5.76%	88.06%	0.64%	7.98%
	Weight Decay	73.47%	1.67%	8.08%	**53.20**%	2.44%	6.97%	88.08%	0.71%	13.74%
	Dropout	73.64%	1.73%	8.23%	53.17%	2.95%	5.96%	87.70%	0.83%	11.52%
	Linear	**75.32**%	0.71%	**4.27**%	53.04%	1.73%	**3.03**%	89.37%	**0.32**%	7.88%
	DARE	74.46%	**0.64**%	4.65%	53.09%	1.86%	3.74%	**89.64**%	0.51%	**7.07**%
	SLERP	75.01%	0.71%	4.34%	53.07%	**1.67**%	3.23%	89.39%	**0.32**%	8.18%
Tool Using Proficiency	Aligned	0.8979%	0.00%	1.22%	0.7280	0.58%	0.00%	0.9357	0.38%	9.09%
	SFT	0.8989	0.83%	3.45%	0.8802	**0.64**%	**0.10**%	0.9369	0.58%	8.08%
	Weight Decay	**0.9282**	1.41%	3.22%	0.8838	0.77%	0.30%	0.9177	0.58%	**8.48**%
	Dropout	0.9269	0.83%	1.92%	**0.8865**	0.83%	0.40%	**0.9514**	0.77%	10.90%
	Linear	0.9266	0.77%	2.44%	0.8793	**0.64**%	0.20%	0.9489	0.13%	9.39%
	DARE	0.9251	**0.45**%	**1.21**%	0.8793	**0.64**%	0.20%	0.9149	**0.06**%	9.39%
	SLERP	0.9266	**0.44**%	1.72%	0.8802	**0.64**%	**0.10**%	0.9152	0.13%	9.19%

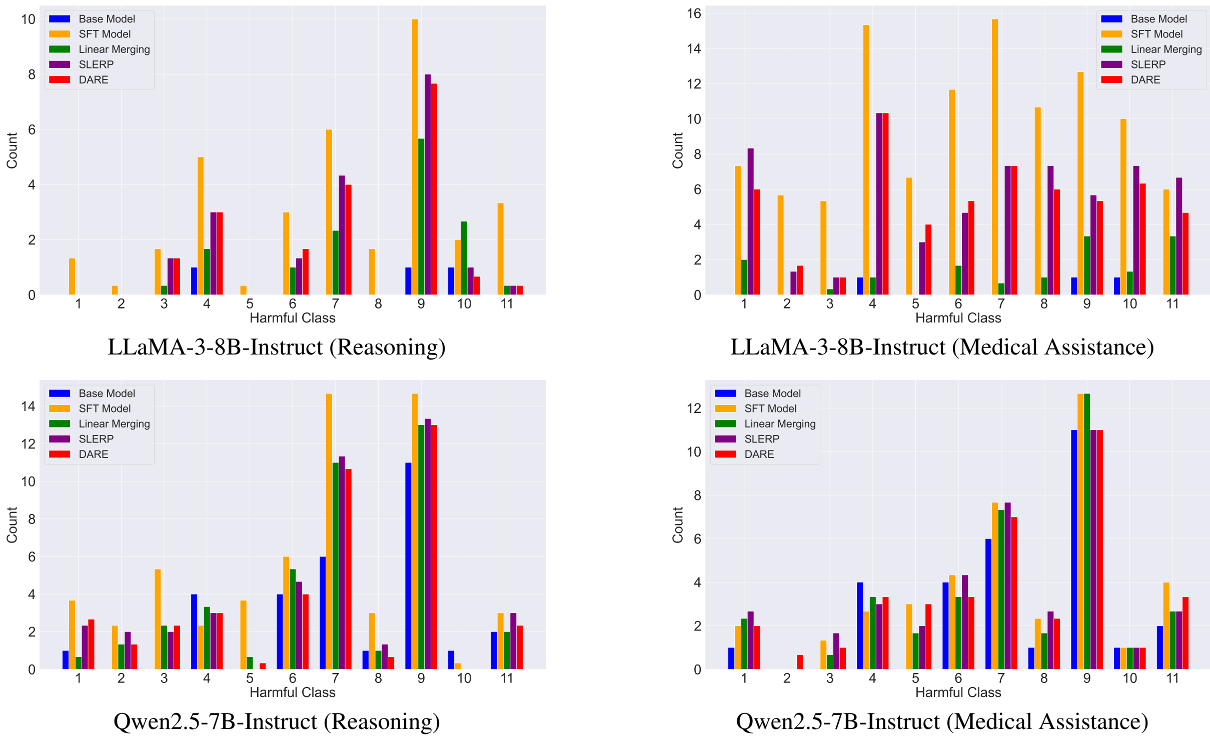

Figure 6: **Safety degradation across categories in the HEx-PHI benchmark.** ASR distributions over 11 harmful categories for LLaMA-3-8B-Instruct and Qwen2.5-7B-Instruct on the Reasoning and Medical Assistance tasks.