Mixup Model Merge: A Geometric Exploration of Task-Vector Space for Decoupled Model Merging

Anonymous ACL submission

Abstract

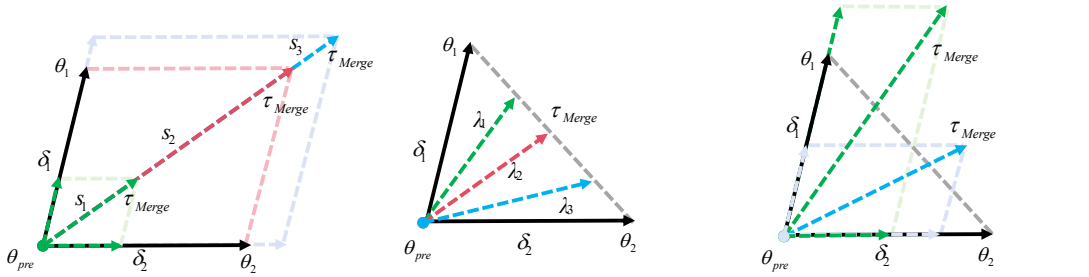
Model merging integrates distinct task-specific capabilities into a unified model without the cost of retraining. We observe that prevailing methods largely focus on resolving parameter-level conflicts while overlooking a macroscopic geometric constraint: they typically assign uniform weights that lock the merging direction, allowing variation only in magnitude. We posit that this rigid constraint prevents the discovery of optimal solutions in the parameter space. In this work, we introduce Mixup Model Merge (M^3), a framework designed to navigate this unexplored geometry by decoupling the exploration of merging direction (via interpolation coefficients) and magnitude (via a scaling factor). By utilizing randomized linear interpolation, M^3 systematically probes the continuous task-vector space to identify low-loss merging configurations. Empirical results on three task-specific LLMs reveal that this simple geometric exploration significantly outperforms complex conflict-resolution baselines. Moreover, M^3 demonstrates superior generalization, substantially enhancing out-of-distribution and adversarial robustness on benchmarks such as LiveBench and PromptBench. Notably, M^3 is orthogonal to sparsification techniques like DARE, unlocking further performance improvements when combined.

1 Introduction

In the rapidly evolving field of Natural Language Processing (NLP), Large Language Models (LLMs) (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2023; Chowdhery et al., 2023) have established themselves as a revolutionary foundation. These models have demonstrated exceptional capabilities across a wide spectrum of tasks (Jiao et al., 2023; Chang et al., 2024b; Nam et al., 2024; Xing, 2024; Guo et al., 2024), fundamentally reshaping the landscape of AI. To adapt these general-purpose models to specific domains, Supervised Fine-Tuning (SFT) and Parameter-Efficient Fine-Tuning (PEFT) have become standard practices (Hu et al., 2021; Ding et al., 2023; Xia et al., 2024). However, the prohibitive computational costs and latency associated with training or maintaining multiple fine-tuned instances (Brown et al., 2020; Chang et al., 2024a) present a significant bottleneck. Consequently, **Model Merging** has emerged as a resource-efficient paradigm to integrate distinct capabilities from multiple homologous models into a single unified parameter set without the need for additional training (Yang et al., 2024; Akiba et al., 2024).

Despite the growing popularity of model merging, existing methodologies predominantly adopt a microscopic perspective, focusing heavily on resolving parameter-level interference. Techniques such as Task Arithmetic (Ilharco et al., 2022), Fisher Merging (Matena and Raffel, 2022), RegMean (Jin et al., 2022), TIES-Merging (Yadav et al., 2023), and DARE (Yu et al., 2024) employ sophisticated heuristics—ranging from Fisher information weighting to delta-parameter pruning—to mitigate conflicts. However, we identify that these approaches typically share a rigid macroscopic geometric constraint: they often assign uniform or scalar merging coefficients to task vectors (Wortsman et al., 2022; Lin et al., 2023; Yadav et al., 2023; Yu et al., 2024). As illustrated in Figure 1(a), this practice effectively **locks the merging direction**, restricting the search space to a fixed linear trajectory where only the magnitude varies. Even linear interpolation methods (Figure 1(b)) are generally confined to the fixed line segment connecting model parameters, limiting the potential to explore the broader parameter space.

We argue that the task vectors of different models represent distinct semantic directions in the high-dimensional parameter space, and their optimal combination is rarely aligned with a simple arithmetic mean. To unlock the full potential of



(a) Same Scaling Factor (b) Linear Interpolation (c) Interpolation Coefficient and Scaling Factor

Figure 1: Geometric view of task vector merging (θ_{pre}: pre-trained model; $\delta_{1,2}$: task vectors). **(a) Standard Merging** restricts solutions to a linear trajectory (varying s). **(b) Linear Interpolation** constrains solutions to the line segment (varying λ). **(c) M^3 (Ours)** decouples λ and s, expanding the search to the conical hull spanned by task vectors.

merging, we propose **Mixup Model Merge (M^3)**, a framework designed to explore the unexplored geometric region between task vectors. As shown in Figure 1(c), M^3 decouples the geometric exploration into two orthogonal components: the **interpolation coefficient** (which determines the merging direction) and the **scaling factor** (which controls the magnitude). Inspired by the randomized interpolation strategy of Mixup in data augmentation (Zhang, 2017), M^3 samples interpolation coefficients from a Beta distribution to sweep through the continuous planar region formed by task vectors, discovering low-loss configurations that rigid merging strategies fail to reach.

We conducted comprehensive experiments on three homologous task-specific LLMs: WizardLM-13B (Xu et al., 2024), WizardMath-13B (Luo et al., 2023), and llama-2-13b-code-alpaca (Chaudhary, 2023), covering instruction following, mathematical reasoning, and code generation. The results demonstrate that the simple geometric exploration enabled by M^3 consistently outperforms complex conflict-resolution baselines. Furthermore, motivated by the connection between Mixup and robustness (Zhang, 2017), we extend our evaluation to out-of-distribution (OOD) and adversarial scenarios using LiveBench (White et al., 2024) and PromptBench (Zhu et al., 2024). Our findings indicate that M^3 significantly enhances the merged models' robustness against distribution shifts and adversarial attacks (Wang et al., 2023; Zhu et al., 2023), proving that macroscopic geometric exploration is a critical factor in effective model merging.

2 Related Works

Model merging is a technique that integrates the parameters of multiple models to create a unified model with enhanced or diverse capabilities (Wortsman et al., 2022; Ilharco et al., 2022; Matena and Raffel, 2022; Jin et al., 2022; Yadav et al., 2023; Yu et al., 2024; Lin et al., 2024). Task arithmetic (Ilharco et al., 2022) leverages task vectors for model merging through arithmetic operations, incorporating a predefined scaling term to weight the contribution of different models. Fisher Merging (Matena and Raffel, 2022) performs parameter merging by applying weights derived from the Fisher information matrix (Fisher, 1922), resulting in more precise parameter integration. TIES-Merging (Yadav et al., 2023) addresses task conflicts by removing low-magnitude parameters, resolving sign disagreements, and merging only the parameters that align with the final agreed-upon sign. In (Yu et al., 2024), it is found that LLMs can enhance their capabilities through model merging. Additionally, it introduces DARE, a method for sparsifying the delta parameters of the model (Ilharco et al., 2022), significantly improving the performance of various model merging techniques. DELLA (Deep et al., 2024) is a novel model merging technique that integrates a new pruning strategy called MAGPRUNE, which samples delta parameters based on their magnitudes. MAGPRUNE demonstrates improvements over existing methods such as DARE and TIES.

3 Methodology

3.1 Geometric Formulation: Decoupling Direction and Magnitude

Standard model merging paradigms typically combine task vectors using uniform or fixed coefficients, implicitly locking the search to a specific linear trajectory, as illustrated in Figure 1(a). We challenge this restriction by formalizing the merging problem through a geometric lens, explicitly

decoupling the merging direction from the magnitude.

Let $\theta_{\text{pre}} \in \mathbb{R}^d$ denote the pre-trained model parameters. For N downstream tasks, we obtain task-specific fine-tuned parameters $\{\theta_{\text{SFT}}^{t_k}\}_{k=1}^N$. The task vector for task k is defined as the displacement induced by fine-tuning:

$$\delta_{t_k} = \theta_{\text{SFT}}^{t_k} - \theta_{\text{pre}}. \tag{1}$$

We posit that the geometry of the optimal merged update δ_M is governed by two orthogonal components:

1. **Direction (Interpolation Coefficients):** The orientation of the update in the subspace spanned by task vectors is determined by a set of interpolation coefficients $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_N\}$, where $\sum \lambda_k = 1$. This defines the normalized direction of the joint adaptation (see Figure 1(b)).

2. **Magnitude (Scaling Factor):** The intensity of the adaptation along the chosen direction is controlled by a global scaling factor $s \in \mathbb{R}^+$.

The final merged model is obtained by traversing this decoupled space:

$$\theta_M = \theta_{\text{pre}} + s \cdot \sum_{k=1}^{N} \lambda_k \delta_{t_k}. \tag{2}$$

As depicted in Figure 1(c), by varying $\boldsymbol{\lambda}$, we sweep through the "cone" of possible directions; by adjusting s, we control the scaling factor to find the optimal intensity of the merged capabilities.

3.2 Geometric Probing: Randomized Exploration via Beta Sampling

To systematically explore this parameter space and validate the importance of geometric diversity, we introduce **Mixup Model Merge (M^3)**. M^3 serves not merely as a merging algorithm, but as a randomized probe to identify optimal geometric configurations that lie beyond the reach of rigid baselines.

Two-Model Exploration via Beta Sampling. For merging two models ($N = 2$), the direction is determined by a single interpolation coefficient $\lambda \in (0, 1)$. To explore diverse directions efficiently, we sample λ from a Beta distribution:

$$\delta_M = \lambda \delta_{t_1} + (1 - \lambda)\delta_{t_2}, \quad \lambda \sim \text{Beta}(\alpha, \alpha). \tag{3}$$

The hyperparameter α shapes the probability distribution for our directional exploration.

- $\alpha < 1$ (Bimodal): Biases exploration towards the axes of individual task vectors, emphasizing task-specific dominance.

- $\alpha > 1$ (Unimodal): Biases exploration towards the arithmetic mean direction, favoring balanced contributions.

- $\alpha = 1$ (Uniform): Provides unbiased exploration of the entire directional sector.

While our framework supports a joint search for the scaling factor s, this study primarily focuses on investigating the contribution of the **merging direction** by exploring λ to reveal the inherent geometric properties of the task-vector space.

3.3 Structured Navigation: Exploiting Asymmetry via P-Series

Empirical Insight: The Asymmetry of Optimal Directions. Extensive exploration reveals a consistent geometric property: optimal merging directions are highly asymmetric. High-performing models rarely reside on the bisector (where λ_k are equal); instead, they are often skewed towards a dominant task. This suggests that effective merging requires a non-uniform allocation of directional influence rather than a simple proportional scaling.

Challenge: Inefficiency of Randomized Sampling in High Dimensions. While Beta sampling is effective for bivariate exploration ($N = 2$), extending randomized probing (e.g., via Dirichlet distribution) to multi-task settings ($n > 2$) becomes computationally prohibitive due to the curse of dimensionality. The probability of randomly hitting the specific "asymmetric pockets" where optimal performance lies diminishes exponentially as the number of tasks increases. Consequently, a more deterministic distribution is required to target these skewed regions efficiently.

Algorithmic Strategy: P-Series Initialization. To address this, we replace randomized sampling with a structured directional search using a decreasing p-series. We define the interpolation coefficients λ_k as:

$$\lambda_k = \frac{k^{-p}}{\sum_{j=1}^{n} j^{-p}}, \quad \text{sorted by task dominance.} \tag{4}$$

Here, the parameter p explicitly controls the "directional skewness" (or concentration) towards dominant tasks. Unlike the Beta distribution which

randomly probes the space, the p-series acts as a navigable compass, allowing us to systematically traverse the gradient of task dominance from uniform ($p = 0$) to highly specific ($p \to \infty$).

3.4 Theoretical Analysis: From Fixed Linear Trajectories to High-Dimensional Cones

We justify the effectiveness of M^3 through the lens of manifold dimensionality and search capacity.

Foundation: The Manifold Hypothesis. Fine-tuned models derived from the same backbone typically reside in a shared, low-loss basin connected by linear paths. This implies that the solution for an optimal merge exists on a continuous manifold containing these task vectors.

Comparison: Search Space Dimensionality. Standard merging methods restrict the search to a fixed linear trajectory defined by the average vector $\bar{\delta} = \frac{1}{N}\sum \delta_k$:

$$\mathcal{S}_{\text{Standard}} = \{\theta_{\text{pre}} + s \cdot \bar{\delta} \mid s \in \mathbb{R}^+\}. \quad (5)$$

This assumes the optimal direction is strictly fixed. However, M^3 decouples the interpolation coefficients $\boldsymbol{\lambda}$ from the scaling factor s, expanding the search space to a higher-dimensional cone:

$$\mathcal{S}_{M^3} = \{\theta_{\text{pre}} + s \cdot \sum \lambda_k \delta_k \mid s \in \mathbb{R}^+, \sum \lambda_k = 1\}. \quad (6)$$

Since $\mathcal{S}_{\text{Standard}} \subset \mathcal{S}_{M^3}$, M^3 theoretically guarantees a solution space that encompasses the standard ray while unlocking access to off-axis regions. This expanded geometry allows M^3 to locate asymmetric optimal points—where task contributions are imbalanced but performance is maximized—that are inaccessible to fixed-direction methods.

4 Experiments

4.1 Experimental Setup

Task-Specific Fine-Tuned LLMs and Datasets. Following the setup in Yu et al. (2024), we utilize three homologous task-specific models fine-tuned from Llama-2-13b (Touvron et al., 2023): WizardLM-13B (Instruction) (Xu et al., 2024), WizardMath-13B (Math) (Luo et al., 2023), and llama-2-13b-code-alpaca (Code) (Chaudhary, 2023). We evaluate their performance on standard benchmarks: AlpacaEval (Li et al., 2023) for instruction following; GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) for mathematical reasoning; and HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) for code generation. See Appendix A for details.

Implementation of Geometric Exploration. To probe the geometric properties of model merging:

- For **Two-Model Merging**, we sweep the merging direction by sampling the interpolation coefficient λ from a Beta distribution Beta(α, α) with $\alpha \in \{0.2, 0.4, 0.5, 1, 2, 3, 5\}$. This allows us to traverse the planar region between task vectors.
- For **Three-Model Merging**, we structure the merging direction using the p-series method ($p \in \{0, 1, 2, 3\}$) to control the skewness towards dominant tasks, coupled with a sweep of the scaling factor $s \in [0.5, 2.0]$.

Unless otherwise specified, we report the performance of the best configuration found on the validation set, comparing it against baselines like Average Merging, Task Arithmetic (Ilharco et al., 2022), TIES-Merging (Yadav et al., 2023), and DARE (Yu et al., 2024).

4.2 Impact of Merging Direction

We first investigate whether expanding the search space from a one-dimensional subspace (fixed direction) to a 2D plane (variable direction + magnitude) yields superior models in the context of pairwise model merging. We integrate M^3 into representative merging methods and report the results for two-model merging in Table 1.

Discovery 1: Optimal Directions are Rarely Bisectors. Standard merging (e.g., Task Arithmetic) implicitly assumes the optimal direction is the arithmetic mean ($\lambda = 0.5$). However, our results contradict this. As shown in Table 1, allowing M^3 to explore off-axis directions consistently unlocks performance gains. For instance, merging Math & Code with Task Arithmetic + M^3 improves MBPP performance by 10.4% over the baseline. Notably, the optimal interpolation coefficients found were often highly asymmetric, effectively "steering" the merged vector closer to the stronger model (Math) while retaining essential features from the weaker one.

Discovery 2: Geometric Correction for Suboptimal Models. Yu et al. (2024) observed that merging WizardMath with the weaker Code-Alpaca model degrades performance. Through our geometric lens, we interpret this as the Code model's

Merging Methods	Models	Use M^3 (Ours)	λ	Instruction Following	Mathematical Reasoning		Code Generating	
				AlpacaEval	GSM8K	MATH	HumanEval	MBPP
-	LM	-	-	45.14	2.20	0.04	36.59	34.00
	Math	-	-	-	64.22	14.02	-	-
	Code	-	-	-	-	-	23.78	27.60
Task Arithmetic	LM & Math	No	-	**45.78**	**66.34**	13.40	-	-
		Yes	0.34	41.65	**66.34**	**13.74**	-	-
	LM & Code	No	-	44.64	-	-	32.93	33.60
		Yes	0.99	**46.64**	-	-	**35.37**	**33.80**
	Math & Code	No	-	-	**64.67**	**13.98**	**8.54**	8.60
		Yes	0.98	-	63.53	13.94	7.93	**19.00**
TIES-Merging	LM & Math	No	-	38.63	14.56	2.12	-	-
		Yes	0.62	**38.73**	**18.57**	**2.48**	-	-
	LM & Code	No	-	41.85	-	-	0.0	0.0
		Yes	0.84	**44.96**	-	-	**25.61**	**30.80**
	Math & Code	No	-	-	64.67	13.68	9.15	**22.60**
		Yes	0.63	-	**64.75**	**14.16**	**9.76**	21.4
DARE	LM & Math	No	-	**49.00**	66.64	13.2	-	-
		Yes	0.38	44.90	**67.32**	**13.74**	-	-
	LM & Code	No	-	41.47	-	-	**35.98**	33.00
		Yes	0.99	**45.20**	-	-	**35.98**	**35.20**
	Math & Code	No	-	-	65.05	10.37	**9.15**	9.80
		Yes	0.98	-	**65.13**	**14.32**	8.54	**18.00**

Table 1: Exploration of the task-vector geometric space: Comparison of merged model performance before and after applying M^3. Bold indicates the higher value within each No/Yes pair (under the same merging method and model combination). The discovered optimal interpolation coefficients (λ) consistently deviate from the standard arithmetic mean ($\lambda = 0.5$), revealing a prevalent asymmetry in the optimal merging direction across various paradigms.

task vector pointing in a "noisy" direction. M^3 effectively performs a geometric correction: by adjusting the interpolation coefficient, it rotates the merged vector away from the noisy component of the Code model while increasing the scaling factor to amplify the effective Math features. This results in a merged model that outperforms both individual experts on 4 out of 5 datasets.

4.3 Navigating the High-Dimensional Cone: Three-Model Merging

Moving beyond pairwise merging, we validate our approach in the more complex 3-task vector space (Instruction, Math, Code). Here, the search space expands from a planar sector to a high-dimensional conical hull. We use the p-series strategy to define the merging direction and a global scalar s for magnitude.

Superiority of Geometric Search. As shown in Figure 2, standard Task Arithmetic (which fixes the direction to the centroid) struggles to balance the three tasks, particularly degrading in Instruction Following (AlpacaEval 6.27%). In contrast, by optimizing the direction (via p) and magnitude (via s), M^3 achieves a significantly better trade-off, boosting AlpacaEval to **7.20%** and GSM8K from 58.52% to **67.55%**. This confirms that the "centroid" of the task vectors is rarely the optimal solution in multi-task scenarios.

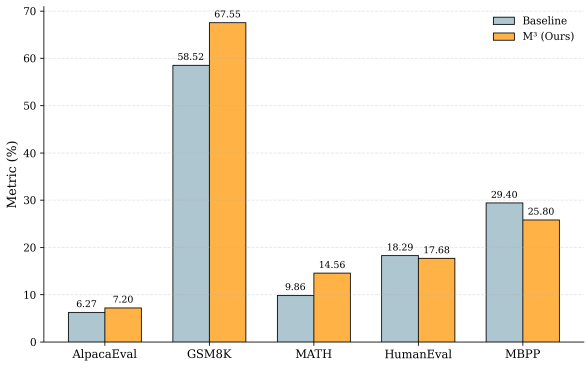


Figure 2: Performance comparison between the fixed-direction baseline (Task Arithmetic) and our proposed M^3 framework on a three-model merge (Instruct, Math, and Code). M^3 identifies an asymmetric merging direction that significantly enhances reasoning capabilities (GSM8K and MATH) while maintaining a robust balance across all tasks.

Impact of Directional Ordering. In high-dimensional space, the order in which we prioritize tasks determines the semantic octant of our search. We hypothesize that the merging direction should align closer to the task with the strongest gradients and highest dominance.

Figure 3 validates this hypothesis. When the dominant direction is set towards Math, the model retains strong mathematical reasoning capabilities. Specifically, prioritizing Math over other tasks Math > Instruct > Code achieves 63.08% on GSM8K, while Math > Code > Instruct achieves 62.90%. In stark contrast, directing the search primarily towards Instruct Instruct > Math > Code leads to catastrophic collapse in mathematical reasoning, with GSM8K dropping to 2.65%.

This result strongly suggests that mathematical reasoning serves as the dominant task manifold in our setup. The merging direction is a primary determinant of success and must preserve access to this critical capability.

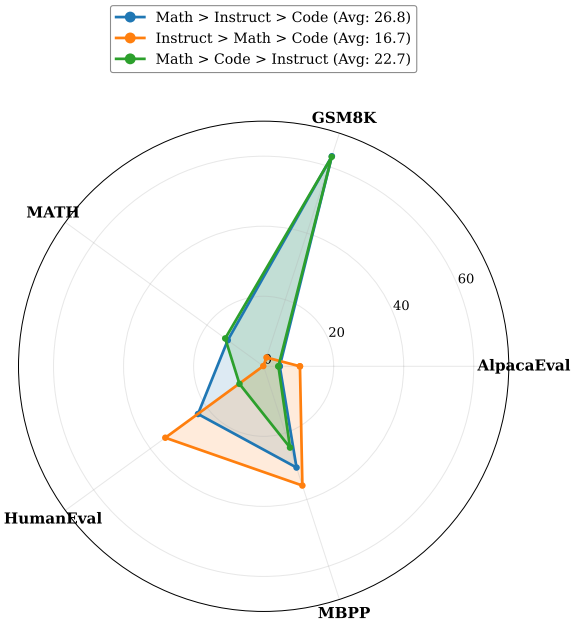


Figure 3: Radar chart comparing three merging directions. Each direction represents a different priority order: mathematical reasoning (Math), instruction following (Instruct), and code generation (Code). The filled areas show performance balance across five evaluation benchmarks.

Decoupling Direction and Magnitude. Finally, we demonstrate the necessity of jointly optimizing both geometric components. Figure 4 shows how performance varies with direction (p) and magnitude (s).

- **Varying Direction (p):** Figure 4a illustrates the weight distribution for different p values. At $p = 0$, weights are uniform (0.333 each). As p increases, Math weight grows rapidly (0.861 at $p = 3$), creating a highly skewed direction.

- **Varying Magnitude (s):** Figure 4b compares the baseline ($p = 0$, $s = 1$) with our optimal configuration ($p = 1$, $s = 1.7$). The optimal point achieves substantial improvements on mathematical tasks (GSM8K: +9.02%, MATH: +4.70%) with minimal degradation elsewhere.

This empirical evidence supports our "Direction + Magnitude" hypothesis: both components must be tuned jointly. The performance peak lies at ($p = 1, s = 1.7$), not at naive averaging ($p = 0, s = 1$) nor at extreme skewing ($p = 3$).

4.4 Synergy with Parameter Sparsification (DARE)

We evaluate the compatibility of M^3 with DARE (Yu et al., 2024), a representative sparsification method designed to resolve parameter interference. As detailed in Appendix Table 7, our experiments reveal two key insights:

M^3 is Effective in Isolation. M^3 generally outperforms DARE when applied separately. For instance, in the Math & Code merging task (MBPP dataset), M^3 achieves a Pass@1 score of 19.0%, significantly surpassing DARE's 9.8% (with a fixed drop rate of 0.2). This indicates that optimizing the interpolation coefficient (λ) to find the correct merging direction is often more critical than parameter pruning alone.

Complementary Mechanisms. Furthermore, combining M^3 with DARE yields additive performance gains. We interpret this synergy through a geometric lens: DARE operates at the microscopic level, reducing the internal noise within task vectors via sparsification; in contrast, M^3 operates at the macroscopic level, optimizing the external alignment (direction and magnitude) of the merged model. Since they address different aspects of model conflict—redundancy versus orientation—M^3 serves as an orthogonal plug-in that can be seamlessly integrated to enhance existing interference-resolution techniques.

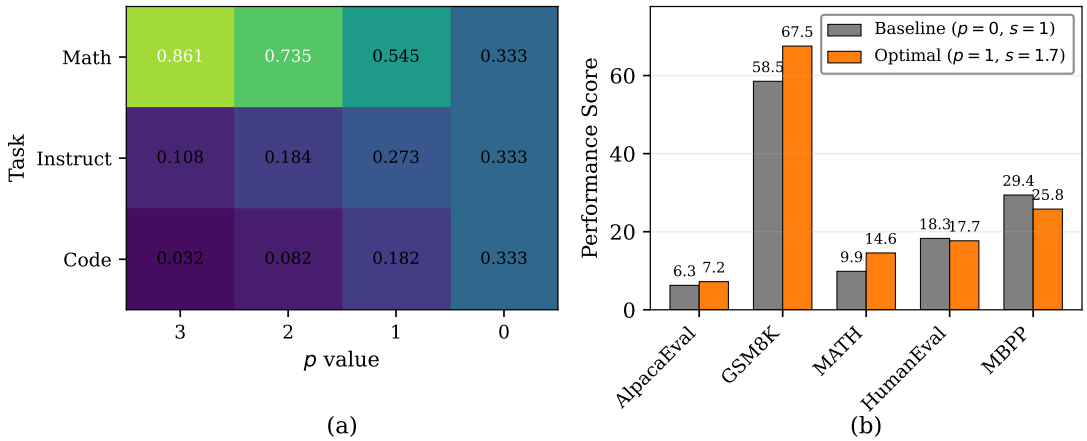


Figure 4: Joint optimization of direction and magnitude is essential. (a) Weight distribution across tasks as p varies. Higher p increases weight disparity toward Math. (b) Optimal configuration ($p = 1$, $s = 1.7$) versus naive averaging ($p = 0$, $s = 1$). The optimal point boosts mathematical reasoning substantially (GSM8K: $58.53\% \rightarrow 67.55\%$) while maintaining performance on other tasks, validating our "Direction + Magnitude" hypothesis.

4.5 Model Robustness

Out-of-Distribution (OOD) Robustness. To ensure the evaluation datasets reflect true OOD scenarios, we select recently released datasets from domains not seen during fine-tuning: Math & Code is evaluated on LiveBench-Instruction, LM & Math on LiveBench-Coding, and LM & Code on LiveBench-TypoFixing. Notably, the test samples in these three datasets are not present in the pre-training corpora of our base models, as they were released significantly after the data cutoff of the underlying backbone.

As shown in Figure 5, M^3 consistently improves OOD performance across all model pairs and merging methods. Specifically, under Task Arithmetic, M^3 yields gains of 1.9%, 1.6%, and 6.0% on the three respective tasks. Similar improvements are observed with TIES-Merging (1.1%, 0.6%, 14.0%). These results highlight M^3's effectiveness in enhancing OOD generalization by exploring the continuous directional space via the interpolation coefficient.

Adversarial Robustness. We evaluate the adversarial robustness of merged models using the StressTest attack from PromptBench (Zhu et al., 2024). Robustness is measured by the Performance Drop Rate (PDR), with a lower PDR indicating stronger robustness.

As demonstrated in Table 2, M^3 notably improves robustness under StressTest attacks. For instance, it reduces PDR from 98.53% to 6.42% for Math & Code on the CoLA dataset, and from 38.04% to 7.68% for LM & Code on SST2. A possible explanation is that standard merging directions (e.g., $\lambda = 0.5$) may align with directions where task vectors conflict, making the model sensitive to perturbations. By adjusting the merging direction and decoupling the interpolation coefficient and scaling factor, M^3 can identify configurations that balance task contributions while maintaining stability.

5 Conclusion

Existing model merging methods often assign the same coefficient to all task vectors, limiting the merged model to a fixed direction and underutilizing complementary capabilities. We propose Mixup Model Merge (M^3), which jointly explores merging direction and step size via randomized linear interpolation in parameter space. Interpolation coefficients sampled from a distribution (e.g., Beta) allow flexible exploration of contribution ratios, and the best merged model is selected on a validation set. Experiments on three task-specific LLMs show that M^3 consistently improves task performance and enhances out-of-distribution and adversarial robustness. M^3 also complements sparsification methods such as DARE. This simple, effective approach provides a general framework for optimizing model merging and can potentially extend to merging with RLHF-aligned models to reduce alignment costs.

Limitations

The current exploration relies on distribution-specific sampling. While this approach is effective

Model	Dataset	Use Mixup	Use Attack	Metric (%)	PDR (%)
Math & Code	SST2	No	No	57.68	38.97
			Yes	35.21	
		Yes	No	<u>86.24</u>	**35.77**
			Yes	55.39	
	CoLA	No	No	45.54	98.53
			Yes	0.67	
		Yes	No	<u>71.72</u>	**6.42**
			Yes	67.11	
LM & Math	SST2	No	No	92.78	**29.05**
			Yes	65.83	
		Yes	No	<u>91.28</u>	34.55
			Yes	59.75	
	CoLA	No	No	79.19	8.84
			Yes	72.20	
		Yes	No	<u>80.54</u>	**4.52**
			Yes	76.89	
LM & Code	SST2	No	No	10.55	38.04
			Yes	6.54	
		Yes	No	<u>73.17</u>	**7.68**
			Yes	67.55	
	CoLA	No	No	74.21	47.42
			Yes	39.02	
		Yes	No	<u>74.78</u>	**31.67**
			Yes	51.10	

Table 2: Adversarial robustness of merged models on SST2 and CoLA with StressTest prompt attacks. Best and second-best results are marked in bold and underlined, respectively.

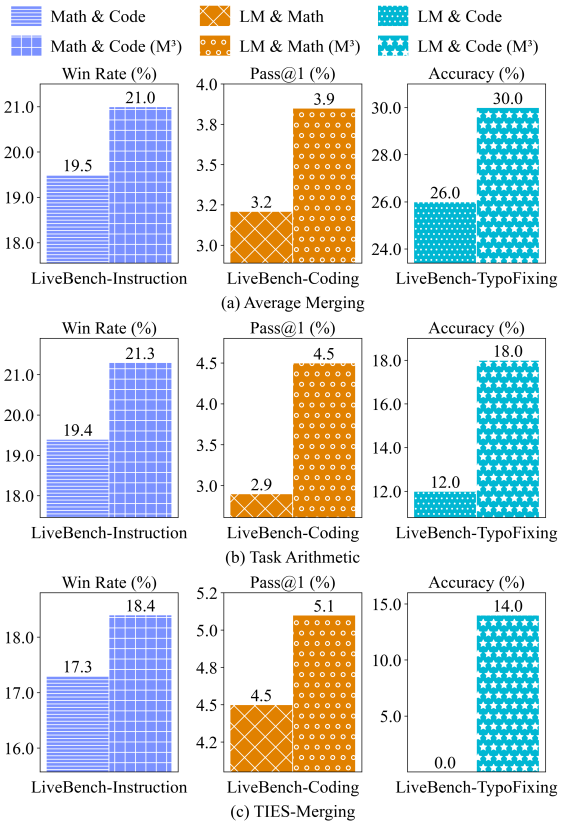


Figure 5: Performance of merged models (Math & Code, LM & Math, and LM & Code) using three model merging methods (Average Merging, Task Arithmetic, and TIES-Merging) on OOD datasets.

for discovering high-performance configurations, investigating more automated search techniques could further enhance efficiency in complex multi-task settings.

Ethical Considerations

Our work contributes to energy-efficient AI development by enhancing model capabilities without the significant computational cost. However, since model merging directly integrates parameters from parent models (e.g., Llama-2 derivatives), the merged models may inherit or potentially amplify existing biases and safety risks present in the source checkpoints.

References

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

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard. *Hugging Face*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yupeng Chang, Yi Chang, and Yuan Wu. 2024a. Balora: Bias-alleviating low-rank adaptation to mitigate catastrophic inheritance in large language models. *arXiv preprint arXiv:2408.04556*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024b. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Sahil Chaudhary. 2023. Code alpaca: An instruction-

following llama model for code generation. *GitHub repository*.

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*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Pala Tej Deep, Rishabh Bhardwaj, and Soujanya Poria. 2024. Della-merging: Reducing interference in model merging through magnitude-based sampling. *arXiv preprint arXiv:2406.11617*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, and 1 others. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Ronald A Fisher. 1922. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 222(594-604):309–368.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Chenlu Guo, Nuo Xu, Yi Chang, and Yuan Wu. 2024. Chbench: A chinese dataset for evaluating health in large language models. *arXiv preprint arXiv:2409.15766*.

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. *arXiv preprint arXiv:2103.03874*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Xing Wang, and Zhaopeng Tu. 2023. Is chatgpt a good translator? a preliminary study. *arXiv preprint arXiv:2301.08745*, 1(10).

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2022. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. *arXiv preprint arXiv:2004.09984*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, and 1 others. 2023. Mitigating the alignment tax of rlhf. *arXiv preprint arXiv:2309.06256*.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, and 1 others. 2024. Mitigating the alignment tax of rlhf. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 580–606.

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*.

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

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. *arXiv preprint arXiv:1806.00692*.

Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. 2024. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13.

R OpenAI. 2023. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5).

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jindong Wang, Xixu Hu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Haojun Huang, Wei Ye, Xiubo Geng, and 1 others. 2023. On the robustness of chatgpt: An adversarial and out-of-distribution perspective. *arXiv preprint arXiv:2302.12095*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

A Warstadt. 2019. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, and 1 others. 2024. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

Tingyu Xia, Bowen Yu, Kai Dang, An Yang, Yuan Wu, Yuan Tian, Yi Chang, and Junyang Lin. 2024. Rethinking data selection at scale: Random selection is almost all you need. *arXiv preprint arXiv:2410.09335*.

Frank Xing. 2024. Designing heterogeneous llm agents for financial sentiment analysis. *ACM Transactions on Management Information Systems*.

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. Resolving interference when merging models. *arXiv preprint arXiv:2306.01708*, 1.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. 2024. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*.

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*.

Hongyi Zhang. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Gong, and 1 others. 2023. Promptrobust: Towards evaluating the robustness of large language models on adversarial prompts. In *Proceedings of the 1st ACM Workshop on Large AI Systems and Models with Privacy and Safety Analysis*, pages 57–68.

Kaijie Zhu, Qinlin Zhao, Hao Chen, Jindong Wang, and Xing Xie. 2024. Promptbench: A unified library for evaluation of large language models. *Journal of Machine Learning Research*, 25(254):1–22.

A Task-Specific Fine-Tuned LLMs and Datasets Details

We conduct model merging experiments using three task-specific LLMs fine-tuned from Llama-2-13b:

- **WizardLM-13B** is an instruction-following model based on Llama-2-13b, designed to improve open-domain instruction-following. Using the Evol-Instruct method (Xu et al., 2024), it generates high-complexity instruction data to reduce human annotation and enhance generalization. The model undergoes supervised fine-tuning with AI-generated data, followed by refinement via RLHF. Evaluation results show that Evol-Instruct-generated instructions outperform human-written ones,

and WizardLM-13B surpasses ChatGPT in high-complexity tasks. In GPT-4 automated evaluation, it achieves over 90% of ChatGPT's performance in 17 out of 29 tasks, demonstrating the effectiveness of AI-evolved instruction fine-tuning (Xu et al., 2024).

- **WizardMath-13B**, optimized from Llama-2-13b, is designed for mathematical reasoning and enhances Chain-of-Thought (CoT) (Wei et al., 2022) capabilities. It uses Reinforcement Learning from Evol-Instruct Feedback to evolve math tasks and improve reasoning. Trained on GSM8K and MATH datasets, it excels in both basic and advanced math problems. In evaluations, WizardMath-Mistral 7B outperforms all open-source models with fewer training data, while WizardMath 70B surpasses GPT-3.5-Turbo, Claude 2, and even early GPT-4 versions in mathematical reasoning tasks.

- **llama-2-13b-code-alpaca** is a code generation model fine-tuned from Llama-2-13b, designed to enhance code understanding and generation. It follows the same training approach as Stanford Alpaca (Taori et al., 2023) but focuses on code-related tasks. The model is fine-tuned with 20K instruction-following code samples generated using the Self-Instruct method (Wang et al., 2022). However, as it has not undergone safety fine-tuning, caution is required when using it in production environments.

We use one dataset to evaluate the instruction-following task:

- **AlpacaEval** (Li et al., 2023) is an LLM-based automated evaluation metric that assesses model performance by testing on a fixed set of 805 instructions and computing the win rate of the evaluated model against a baseline. The evaluation process involves an LLM-based evaluator that compares the responses and determines the probability of preferring the evaluated model. In this paper, we use AlpacaEval 2.0 (Dubois et al., 2024). To reduce costs, we use chatgpt_fn for evaluation.

We use two dataset to evaluate the mathematical reasoning task:

- **GSM8K** is a dataset of 8.5K high-quality, linguistically diverse grade school math word problems, designed to evaluate the multi-step mathematical reasoning abilities of large language models. It consists of 7.5K training problems and 1K test problems. In this paper, we use the 1K test set for evaluation (Cobbe et al., 2021).

- **MATH** is a dataset containing 12,500 competition-level math problems, designed to evaluate and enhance the problem-solving abilities of machine learning models. It consists of 7,500 training problems and 5,000 test problems. We use the 5,000 test set for evaluation (Hendrycks et al., 2021).

We used two dataset to evaluate the code generation task:

- **HumanEval** is a dataset consisting of 164 hand-written programming problems, designed to evaluate the functional correctness of code generation models. Each problem includes a function signature, docstring, function body, and unit tests. The dataset tests models' language comprehension, reasoning, and algorithmic abilities (Chen et al., 2021).

- **MBPP** is a dataset containing 974 programming problems designed to evaluate a model's ability to synthesize Python programs from natural language descriptions. The problems range from basic numerical operations to more complex tasks involving list and string processing. The test set consists of 500 problems, which are used for evaluation in this paper (Austin et al., 2021).

B Out-of-Distribution Dataset Selection Details

LiveBench (White et al., 2024) is a dynamic benchmark for large language models, featuring frequently updated questions and diverse tasks. To assess OOD robustness, we evaluate math & code, LM & math, and LM & code models using instruction following (LiveBench-Instruction), coding (LiveBench-Coding), and language comprehension (LiveBench-TypoFixing) category in LiveBench, respectively, deliberately avoiding the fine-tuning domains of the merged fine-tuned models. These tasks were released after November 2023, whereas

WizardLM-13B, WizardMath-13B, and llama-2-13b-code-alpaca were all introduced earlier. Furthermore, their shared Llama-2-13b backbone was trained on data only up to July 2023. Consequently, these factors collectively ensure that the evaluation remains OOD in the temporal dimension.

When assessing the OOD robustness of LM & Code using the Language Comprehension category in LiveBench, only the typo-fixing task is considered. This decision is based on the fact that LiveBench is highly challenging, and the merged model performs poorly on other tasks in this category, with accuracy close to zero, rendering the evaluation results inconclusive and uninformative.

Finally, we acknowledge the limitations of these datasets. For large models like Llama-2-13b, identifying truly OOD datasets is difficult, as their training data likely covers similar distributions. These datasets are better described as "out-of-example", representing instances not explicitly seen during training. As discussed in (Wang et al., 2023), distribution shifts can occur across domains and time. While Llama-2-13b may have been trained on datasets for tasks like instruction-following, coding, and language comprehension, the datasets we selected remain valuable for OOD evaluation by capturing temporal shifts, providing insights into robustness over time.

C Adversarial Robustness Evaluation Experiments Setting Details

PromptBench (Zhu et al., 2024) is a unified library designed for evaluating LLMs, providing a standardized and extensible framework. It includes several key components such as prompt construction, prompt engineering, dataset and model loading, adversarial prompt attacks, dynamic evaluation protocols, and analysis tools.

We use the Adversarial Prompt Attacks module in PromptBench aims to evaluate the robustness of LLMs against adversarial prompts. We employ three methods to perform adversarial attacks on prompts to evaluate the adversarial robustness of the merged models: DeepWordBug (Gao et al., 2018), BERTAttack (Li et al., 2020), and StressTest (Naik et al., 2018), representing Character-level, Word-level, and Sentence-level attacks, respectively.

- **DeepWordBug** introduces subtle character-level perturbations (e.g., adding, deleting, or replacing characters) to words in text to deceive language models. It aims to evaluate a model's robustness against small typographical errors that may alter the model's performance without being easily detected.

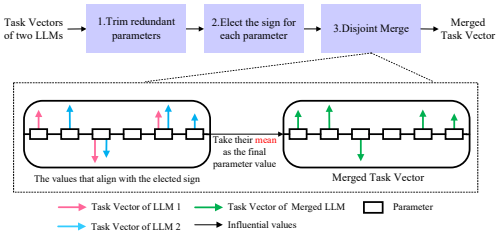
- **BERTAttack** manipulates text at the word level by replacing words with contextually similar synonyms to mislead large language models. This method tests the model's ability to maintain accuracy despite small lexical changes that might alter the meaning of the input.

- **StressTest** appends irrelevant or redundant sentences to the end of a prompt to distract and confuse language models. It assesses the model's ability to handle extraneous information and maintain accuracy when faced with unnecessary distractions.

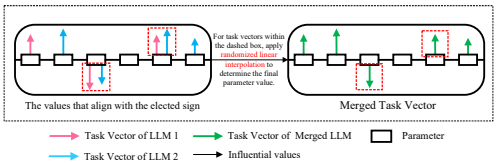
The evaluation is conducted on the Sentiment Analysis dataset (SST2 (Socher et al., 2013)) and the Grammar Correctness dataset (CoLA (Warstadt, 2019)):

- **SST2** (Socher et al., 2013): A sentiment analysis dataset designed to assess whether a given sentence conveys a positive or negative sentiment.

- **CoLA** (Warstadt, 2019): A dataset for grammar correctness, where the model must de-



(a) The operational steps of TIES-Merging.



(b) After introducing M^3, the Disjoint Merge step in the TIES-Merging procedure.

Figure 6: The difference between M^3 and the original TIES-Merging is that, in the Disjoint Merge step, when two task vectors are retained for a given parameter, the mean of the task vectors is replaced by a random linear interpolation, while the other operations remain unchanged.

Merging Methods	Search Ranges of Hyperparameters
Task Arithmetic	Scaling term for merging model parameters: [0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
TIES-Merging	Scaling term for merging model parameters: [0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
	Ratio for retaining parameters with the largest-magnitude values: [0.5, 0.7, 0.9]

Table 3: Hyperparameter search ranges for model merging methods.

Merging Method	Model	Hyperparameter Values
TIES-Merging	LM & Math	scaling_term=0.5, retain_ratio=0.5
	LM & Code	scaling_term=1.0, retain_ratio=0.7
	Math & Code	scaling_term=1.0, retain_ratio=0.5

Table 4: Hyperparameter settings in TIES-Merging.

termine whether a sentence is grammatically acceptable.

D Hyperparameter Setting Details in Model Merging Methods

Table 3 presents the hyperparameter search ranges for the model merging methods. For Task Arithmetic and TIES-Merging, the scaling terms are selected from $[0.5, 1.0]$, while in TIES-Merging, the retain ratio for the largest-magnitude parameters is chosen from $[0.5, 0.7, 0.9]$. In contrast, the Average Merging method does not require any hyperparameters.

Table 4 presents the optimal hyperparameter settings for the TIES-Merging model merging method obtained through searching. These settings are further applied to model merging experiments involving M^3 and DARE.

E Performance Drop Rate (PDR) for Adversarial Robustness

The adversarial robustness is evaluated using the Performance Drop Rate (PDR) (Zhu et al., 2023), which is defined as follows:

$$\text{PDR} = \frac{\text{Metric}_{\text{no attack}} - \text{Metric}_{\text{attack}}}{\text{Metric}_{\text{no attack}}}, \quad (7)$$

where $\text{Metric}_{\text{no attack}}$ denotes the performance metric without any prompt attack, and $\text{Metric}_{\text{attack}}$ represents the performance metric under the prompt attack. A smaller PDR indicates stronger adversarial defense against prompt attacks, implying better adversarial robustness.

F Additional Experimental Results on Adversarial Robustness

All the merged models are obtained using the Task Arithmetic method. Table 5 presents the detailed experimental results of the adversarial robustness of merged models on the SST2 and CoLA datasets applying the DeepWordBug prompt attack method. Table 6 presents the detailed experimental results of the adversarial robustness of merged models on the SST2 and CoLA datasets applying the BERTAttack prompt attack method.

G Detailed Introduction to DARE

DARE (Drop And REscale) (Yu et al., 2024) is a model sparsification method designed to reduce the redundancy of delta parameters in fine-tuned models while preserving their task-specific capabilities. In SFT, model parameters are optimized to unlock abilities for specific tasks, with the difference between fine-tuned and pre-trained parameters referred to as delta parameters.

However, studies have shown that delta parameters are often highly redundant. DARE addresses this redundancy by randomly dropping a proportion p of delta parameters (referred to as the drop rate) and rescaling the remaining ones by a factor of $1/(1-p)$. This simple yet effective approach enables DARE to eliminate up to 99% of delta parameters with minimal impact on model performance, particularly in large-scale models, and it can be applied using only CPUs.

Beyond sparsification, DARE serves as a versatile plug-in for merging multiple homologous fine-tuned models (fine-tuned from the same base model) by reducing parameter interference. When combined with existing model merging techniques such as Average Merging, Task Arithmetic, and TIES-Merging, DARE facilitates the merging of models while retaining or even enhancing task performance across multiple benchmarks. This effect is especially pronounced in decoder-based LMs, where DARE boosts task generalization.

Experiments on AlpacaEval, GSM8K, and MBPP reveal that the merged LM has the potential to outperform any individual source LM, presenting a significant new discovery. Notably, the 7B model obtained through DARE merging, Super-Mario v2, ranks first among models of the same scale on the Open LLM Leaderboard (Beeching et al., 2023). These improvements were achieved without the need for retraining, positioning DARE

Model	Dataset	Use Mixup	Use Attack	Metric (%)	PDR (%)
Math & Code	SST2	No	No	57.68	11.73
			Yes	50.92	
		Yes	No	78.21	37.10
			Yes	49.20	
	CoLA	No	No	72.87	56.97
			Yes	31.35	
		Yes	No	74.02	58.94
			Yes	30.39	
LM & Math	SST2	No	No	92.78	2.60
			Yes	90.37	
		Yes	No	91.28	3.77
			Yes	87.84	
	CoLA	No	No	79.19	4.96
			Yes	75.26	
		Yes	No	80.54	1.07
			Yes	79.67	
LM & Code	SST2	No	No	10.55	98.91
			Yes	0.11	
		Yes	No	73.17	97.65
			Yes	1.72	
	CoLA	No	No	74.21	8.79
			Yes	67.69	
		Yes	No	73.922	11.15
			Yes	65.68	

Table 5: Adversarial robustness of merged models on the SST2 and CoLA datasets when executing the DeepWordBug prompt attack method.

as an efficient and resource-friendly solution for model merging.

H Integrating M^3 into the TIES-Merging Framework

The integration of M^3 into TIES-Merging follows a hierarchical logic: resolving microscopic conflicts first, followed by macroscopic directional exploration. As illustrated in Figure 6, after the standard TIES steps of trimming low-magnitude parameters and resolving sign disagreements, we obtain the refined task vectors for the models to be merged.

Instead of using a fixed scaling or averaging approach, M^3 is applied to these resolved vectors to identify the optimal **interpolation coefficient** (λ). Specifically, for parameters preserved in both task vectors, we perform a weighted interpolation to determine the merged values. For parameters retained in only one of the models, we preserve their original values to maintain task-specific knowledge. This integration demonstrates that M^3 acts as a flexible geometric probe that can be seamlessly combined with conflict-resolution methods, ensuring that the merged model not only avoids parameter interference but also aligns with the most performant direction in the task-vector space.

Model	Dataset	Use Mixup	Use Attack	Metric (%)	PDR (%)
Math & Code	SST2	No	No	57.68	13.92
			Yes	49.66	
		Yes	No	78.21	4.11
			Yes	75.00	
	CoLA	No	No	45.54	13.47
			Yes	39.41	
		Yes	No	71.72	17.25
			Yes	59.35	
LM & Math	SST2	No	No	92.78	2.22
			Yes	90.71	
		Yes	No	91.28	0.0
			Yes	91.28	
	CoLA	No	No	79.19	12.00
			Yes	69.70	
		Yes	No	80.54	5.83
			Yes	75.84	
LM & Code	SST2	No	No	10.55	95.65
			Yes	0.46	
		Yes	No	73.17	55.02
			Yes	32.91	
	CoLA	No	No	74.21	7.24
			Yes	68.84	
		Yes	No	73.92	7.52
			Yes	68.36	

Table 6: Adversarial robustness of merged models on the SST2 and CoLA datasets when executing the Bertattack prompt attack method.

Merging Methods	Models	Use M^3 (Ours)	Use DARE	Instruction Following	Mathematical Reasoning		Code Generating	
				AlpacaEval	GSM8K	MATH	HumanEval	MBPP
Average Merging	LM & Math	Yes	No	**44.40**	66.26	13.80	-	-
		No	Yes	44.22	**66.57**	12.96	-	-
		Yes	Yes	43.53	**66.57**	**14.12**	-	-
	LM & Code	Yes	No	**43.91**	-	-	**37.20**	34.40
		No	Yes	38.81	-	-	31.71	32.40
		Yes	Yes	40.31	-	-	36.59	**37.00**
	Math & Code	Yes	No	-	63.61	**14.02**	8.54	19.20
		No	Yes	-	56.18	10.28	6.10	7.80
		Yes	Yes	-	**64.97**	13.54	**9.76**	**21.20**
Task Arithmetic	LM & Math	Yes	No	41.65	66.34	**13.74**	-	-
		No	Yes	**49.00**	66.64	13.02	-	-
		Yes	Yes	44.90	**67.32**	**13.74**	-	-
	LM & Code	Yes	No	**46.64**	-	-	35.37	33.80
		No	Yes	41.47	-	-	**35.98**	33.00
		Yes	Yes	45.20	-	-	**35.98**	**35.20**
	Math & Code	Yes	No	-	63.53	13.94	7.93	**19.00**
		No	Yes	-	65.05	13.96	**10.37**	9.80
		Yes	Yes	-	**65.13**	**14.32**	8.54	18.00
TIES-Merging	LM & Math	Yes	No	38.73	18.57	2.48	-	-
		No	Yes	37.92	18.04	2.34	-	-
		Yes	Yes	**39.93**	**19.26**	**2.82**	-	-
	LM & Code	Yes	No	44.96	-	-	25.61	30.80
		No	Yes	43.13	-	-	0.0	0.0
		Yes	Yes	**45.65**	-	-	**26.83**	**33.20**
	Math & Code	Yes	No	-	64.75	14.16	9.76	21.4
		No	Yes	-	**64.82**	13.88	**10.37**	**23.60**
		Yes	Yes	-	64.75	**14.78**	9.15	19.60

Table 7: Comparison of our method M^3 and DARE. The best and second-best results are marked in bold and underlined fonts.