Sens-Merging: Sensitivity-Guided Parameter Balancing for Merging Large Language Models

Shuqi Liu[1,2], **Han Wu**[2,†], **Bowei He**[1], **Xiongwei Han**[2], **Mingxuan Yuan**[2], **Linqi Song**[1,†]
[1] Department of Computer Science, City University of Hong Kong
[2] Huawei Noah's Ark Lab
{shuqiliu4-c, boweihe2-c}@my.cityu.edu.hk
wu.han1@huawei.com
linqi.song@cityu.edu.hk

Abstract

Recent advances in large language models have led to numerous task-specialized fine-tuned variants, creating a need for efficient model merging techniques that preserve specialized capabilities while avoiding costly retraining. While existing task vector-based merging methods show promise, they typically apply uniform coefficients across all parameters, overlooking varying parameter importance both within and across tasks. We present Sens-Merging, a sensitivity-guided coefficient adjustment method that enhances existing model merging techniques by operating at both task-specific and cross-task levels. Our method analyzes parameter sensitivity within individual tasks and evaluates cross-task transferability to determine optimal merging coefficients. Extensive experiments on Mistral 7B and LLaMA2-7B/13B models demonstrate that Sens-Merging significantly improves performance across general knowledge, mathematical reasoning, and code generation tasks. Notably, when combined with existing merging techniques, our method enables merged models to outperform specialized fine-tuned models, particularly in code generation tasks. Our findings reveal important trade-offs between task-specific and cross-task scalings, providing insights for future model merging strategies.

1 Introduction

The rapid advancement of large language models (LLMs) has significantly enhanced performance across a diverse range of tasks (Touvron et al., 2023; Zhao et al., 2023). As these models continue to be fine-tuned for specialized domains, the necessity to merge these specialized models into a unified framework becomes increasingly critical (Yang et al., 2024; Goddard et al., 2024). While multi-task learning has been proposed as a solution, it incurs substantial training costs and requires simultaneous access to data and labels for all tasks (Sanh et al., 2022; Fifty et al., 2021). Recently, researchers have developed parameter-level model merging methods that not only comply with data privacy regulations but also improve efficiency by eliminating the need for retraining (Yadav et al., 2023; Yu et al., 2024b).

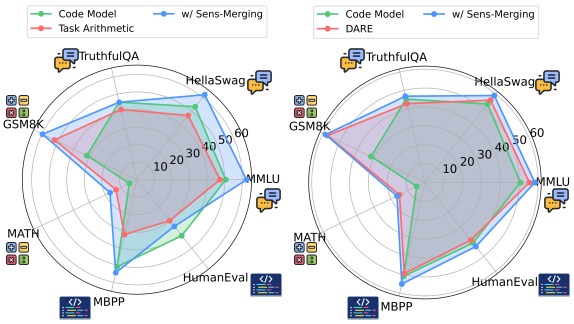

Figure 1: Sens-Merging functions as a plug-and-play enhancement to existing task vector-based merging techniques. Notably, when integrated with DARE, it surpasses even specialized code models in code generation.

In the context of model merging, task vectors (Ilharco et al., 2023a) have emerged as a powerful component for representing task-specific capabilities. These vectors, defined as the differences between parameter values before and after fine-tuning, enable effective integration of specialized knowledge from different models. While task vector-based merging methods (Yadav et al., 2023; Yu et al., 2024b) have shown promising results, their reliance on uniform coefficients for each task and parameter limits their potential effectiveness. This uniformity implies that every task and every parameter is treated with equal importance during the merging process. Consequently, it overlooks the fact that parameters within each layer demonstrate varying levels of importance for specific tasks, and parameters from different tasks contribute distinctly during the merging process.

†Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 19243–19255

To address these challenges, we propose Sens-Merging, a sensitivity-guided merging coefficient adjustment method that functions as a plug-and-play enhancement to existing task vector-based merging techniques. Our method operates at two levels: within individual tasks and across different tasks, allowing for fine-grained control over parameter importance. Within each task-specific model, we perform parameter sensitivity analysis to highlight critical layers that significantly impact performance. Concurrently, across different tasks, we conduct task sensitivity analysis to prioritize models that enhance the performance of others. By combining these two factors, we derive the final merging coefficients, which are then applied to merge the corresponding layers. Figure 1 highlights how Sens-Merging enhances existing task-vector techniques like Task Arithmetic (Ilharco et al., 2023b) and DARE (Yu et al., 2024b). Notably, when combined with DARE, Sens-Merging enables merged models to outperform specialized fine-tuned models, particularly in code generation.

To empirically demonstrate the effectiveness of Sens-Merging, we conduct extensive experiments by combining it with existing model merging approaches. We merged three widely adopted fine-tuned models—specializing in general knowledge (Chat), mathematical reasoning (Math), and code generation (Code)—derived from the LLaMA2-7B/13B and Mistral 7B families. The integration of our Sens-Merging not only improves baseline merging performance but enables merged models to surpass individual fine-tuned models. Notably, when merging Code model with Math and Chat models using Sens-Merging, it achieves superior performance on coding tasks compared to code-specific fine-tuning alone. These results indicate that model merging can effectively address the challenges of training a single model for complex tasks by integrating the specialized capabilities of multiple fine-tuned models.

To sum up, our contributions include: (1) We propose a novel model merging coefficient determination method based on both task-specific and cross-task sensitivity analysis. (2) Through comprehensive evaluations, we validate that our proposed method enhances model merging performance across various domains. (3) We empirically demonstrate that different task-specific models contribute unequally to model merging, and parameter importance varies across different layers within each model. (4) We validate that each scaling approach presents distinct trade-offs: task-specific scaling excels in specialized domains like mathematics but offers limited general benefits, while cross-task scaling achieves broader performance gains at the cost of peak task-specialized performance.

2 Related Work

Modeling merging (Yang et al., 2024; Goddard et al., 2024), as a complementary approach to training-based methods, has the capability to integrate multiple task-specialized models into a unified one (Wortsman et al., 2022; Stoica et al., 2024; Ainsworth et al., 2023; Yu et al., 2024b), to improve model performance on individual tasks by merging checkpoints without requiring additional training (Yadav et al., 2023; Ilharco et al., 2023b), and to alleviates the issue of catastrophic forgetting (Alexandrov et al., 2024). According to whether the based models are in same architecture, the model merging methods can be divided into heterogeneous model merging and homogeneous model merging.

Heterogeneous Model Merging. A brunch of work (Avrahami et al., 2022; Nguyen et al., 2023) attempts to perform architecture transformation before merging, aiming to transform multiple models with different architectures into a unified one. However, these approaches often rely on the learning process to align the models, which can potentially degrade their performance. Recent research in this direction often builds upon the concept of mode connectivity (Freeman and Bruna, 2017; Frankle et al., 2020; Tatro et al., 2020), which suggests the existence of a connected path between multiple local minima of models, along which the loss remains nearly constant. Furthermore, Entezari et al. (2022) revealed that models permuted to the same loss basin can be merged by averaging their weights. Following these intuitions, more recent works (Ainsworth et al., 2023; Jordan et al., 2023; Stoica et al., 2024) focus on permutation strategies to achieve better heterogeneous model merging.

Homogeneous Model Merging. Task-specific models initialized from the same pre-trained model can often be merged without considering permutation symmetry (Wortsman et al., 2022; Ilharco et al., 2023b). One of the most straightforward approaches to model merging is to directly weighted average the parameters of base models (Shoemake, 1985; Wortsman et al., 2022). However, the per-

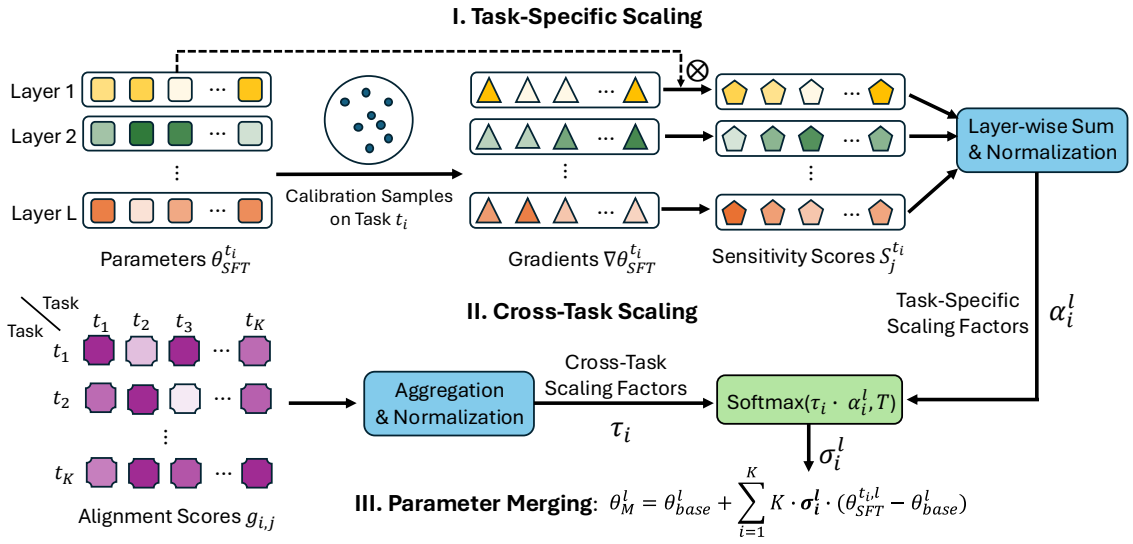

Figure 2: Overall framework of our Sens-Merging method. Sens-Merging adjusts layer-wise scaling coefficients for task-specialized fine-tuned models through two mechanisms: task-specific scaling and cross-task scaling.

formance of simple average merging is often suboptimal, as task-specific features are typically not uniformly distributed. Task Arithmetic (Ilharco et al., 2023b) enhances the merging process by introducing task vectors, suggesting that simple arithmetic operations on these vectors can effectively edit models and produce a merged model. Building on the concept of task vectors, both DARE (Yu et al., 2024b) and Ties (Yadav et al., 2023) employ pruning-then-scaling methods to merge task vectors, based on the assumption that not all parameters contribute equally to the final performance. This also aligns with our perspective. While DARE is primarily designed for merging models with minor parameter changes, WIDEN (Yu et al., 2024a) aims at merging models with significant parameter shifts by disentangling weight components.

Another line of research on model merging leverages information derived from the activations of training data. For example, Matena and Raffel (2022) suggested a probability-space approach, which uses the Fisher information matrix to identify the importance of model parameters and proposed Fisher Merging. Jin et al. (2023) introduced RegMean, a data-less merging method that merges models in parameter space by solving a linear system constructed from data and model parameters. We posit that activations play a crucial role in identifying the key parameters within task vectors relevant to downstream tasks. To this end, we propose a novel sensitivity-guided activation method to facilitate more effective merging of key features.

3 Methodology

Our Sens-Merging method combines two levels of sensitivity analysis: layer-wise analysis within individual models and cross-task analysis across different models to achieve a balanced parameter competition. For layer-wise analysis, we compute sensitivity scores using gradient information from calibration datasets. For cross-task analysis, we evaluate model alignment through logits comparison. These two components determine the final merging coefficients used to merge corresponding layers into a unified model, as shown in Figure 2.

3.1 Preliminary

Considering K task-specialized fine-tuned models $\{\theta_{\text{SFT}}^{t_1}, \ldots, \theta_{\text{SFT}}^{t_K}\}$ derived from a common pre-trained backbone θ_{PRE}, model merging aims to merge them into a single model θ_M that can effectively handle all tasks simultaneously. The task-specific capabilities of each fine-tuned model are captured by task vectors, defined as the difference between the fine-tuned parameters and the pre-trained backbone:

$$\delta_{t_k} = \theta_{\text{SFT}}^{t_k} - \theta_{\text{PRE}}, \quad \text{for } k \in \{1, \ldots, K\}.$$

Task vector-based merging aggregates these task vectors to construct a merged model:

$$\theta_{\text{M}} = \theta_{\text{PRE}} + \sum_{k=1}^{K} \lambda * \delta_{t_k}.$$

where the coefficient λ represents the importance of each merged task vector.

3.2 Task-Specific Scaling

To accurately balance the parameters within individual task models, we conduct layer-wise sensitivity analysis by measuring each layer's contribution to model performance through aggregating parameter sensitivities within that layer.

Parameter Sensitivity. We define parameter sensitivity as the change in loss when setting that parameter to zero. A parameter is considered highly sensitive if zeroing it results in a significant loss increase. For a fine-tuned model with parameters $\theta_{\text{SFT}}^{t_i} = [\theta_1, ..., \theta_N]$, where N represents the total number of parameters, the j-th parameter can be expressed as $\theta_j^{t_i} = [0, ..., \theta_j, ..., 0]$. With gradients of the loss relative to $\theta_{\text{SFT}}^{t_i}$ represented as $\nabla_{\theta_{\text{SFT}}^{t_i}} L$, the sensitivity of the j-th parameter for a specific sample x_k from task t_i is determined as:

$$S_{j,k}^{t_i} = |(\theta_j^{t_i})^\top \nabla_{\theta_{\text{SFT}}^{t_i}} L(x_k)| \tag{1}$$

The rationale behind this sensitivity definition stems from the first-order Taylor expansion of $L(x_k)$ relative to θ_j. In essence, $S_{j,k}^{t_i}$ provides an approximation for how the loss might change in the absence of θ_j:

$$(\theta_j^{t_i})^\top \nabla_{\theta_{\text{SFT}}^{t_i}} L(x_k) \approx L(\theta_{\text{SFT}}^{t_i}) - L(\theta_{\text{SFT}}^{t_i} - \theta_j^{t_i}) \tag{2}$$

To estimate the parameter sensitivity $S_j^{t_i}$ for task t_i, we randomly sample m instances from the task training set as calibration samples. The final sensitivity score $S_j^{t_i}$ aggregates the individual sensitivities across all sampled instances: $S_j^{t_i} = \sum_{k=1}^{m} S_{j,k}^{t_i}$.

Layer-Wise Sensitivity and Normalization. The layer-wise sensitivity s_i^l is then calculated by summing the sensitivities of all parameters within each layer, thereby reflecting each layer's overall contribution to the model's performance. To allow for meaningful comparisons of these importance scores across different models, we apply L_2 normalization to the sensitivities of all layers. Consequently, the task-specific sensitivity scaling factors α_i^l are defined as:

$$s_i^l = \sum_{j \in \mathcal{P}_l} S_j^{t_i}, \quad \alpha_i^l = \frac{s_i^l}{\|\mathbf{s}_i\|_2} \tag{3}$$

where $\mathcal{P}_l$ denotes the set of parameters in layer l, and L is the total number of layers in the model.

3.3 Cross-Task Scaling

While task-specific sensitivity focuses on the importance of layers within individual tasks, it is equally essential to evaluate how each task-specific model influences other tasks during the merging process. Cross-task sensitivity captures the interdependencies and shared representations between different tasks, ensuring that the merged model benefits from common features and decision-making processes.

The measurement of cross-task influence begins with evaluating logits alignment between different task-specific models. Specifically, for calibration samples from task t_j, we compute the alignment score between model $\theta_{\text{SFT}}^{t_i}$ and the expert model for task t_j, $\theta_{\text{SFT}}^{t_j}$, using the L_2 distance between their output logits:

$$g_{i,j} = ||f_{\theta_{\text{SFT}}^{t_i}}(x_k^j) - f_{\theta_{\text{SFT}}^{t_j}}(x_k^j)||_2 \tag{4}$$

where $f_\theta(x)$ denotes the output logits of model θ for input x, and $||\cdot||_2$ represents L_2 distance. This alignment score quantifies how closely the predictions of model $\theta_{\text{SFT}}^{t_i}$ match those of the expert model for task $\theta_{\text{SFT}}^{t_j}$, providing insight into the degree of shared knowledge and representational similarity between tasks. To obtain a comprehensive measure of cross-task sensitivity for a specific task model $\theta_{\text{SFT}}^{t_i}$, we aggregate the alignment scores across all other tasks. This aggregation process involves computing the normalized alignment:

$$\tau_i = \sum_{i=1, i \neq j}^{K} g_{i,j}, \quad \tau_i = 1 - \frac{\tau_i}{\|\boldsymbol{\tau}\|_1} \tag{5}$$

The resulting cross-task scaling factor τ_i serves as a crucial metric that quantifies model $\theta_{\text{SFT}}^{t_i}$'s ability to transfer knowledge across tasks. Higher values of τ_i indicate superior cross-task generalization capabilities, suggesting that the model has learned robust representations that are valuable across multiple tasks. Conversely, lower values of τ_i reflect greater task-specific specialization, indicating that the model's features are more narrowly focused on its primary task.

3.4 Integration with Merging Methods

Our Sens-Merging method combines task-specific scaling factor α_i^l and the cross-task scaling factor τ_i into a plug-and-play module, which can be seamlessly integrated with existing task vector-based model merging methods. To effectively combine

these sensitivity factors, we employ a two-step process. First, we multiply the task-specific scaling factor α_i^l with the cross-task scaling factor τ_i to capture both task-specific and cross-task importance. Then, we apply a softmax function with temperature T to normalize these products and obtain the final scaling coefficients:

$$\sigma_i^l = \text{Softmax}(\tau_i \cdot \alpha_i^l, T) \tag{6}$$

The final step involves computing the merged model parameters θ_M^l for each layer l. We start with the base model parameters θ_{base}^l and incorporate weighted contributions from all K fine-tuned models. The contribution of each task-specific model is scaled by its normalized coefficient σ_i^l and multiplied by K to preserve the magnitude of updates:

$$\theta_M^l = \theta_{\text{base}}^l + \sum_{i=1}^{K} K \cdot \sigma_i^l \cdot (\theta_{\text{SFT}}^{t_i,l} - \theta_{\text{base}}^l) \tag{7}$$

4 Experiments

Baselines. We evaluate the effectiveness of our Sens-Merging method by comparing it against both individual task-specific models and several established model-merging techniques, including Task Arithmetic, Ties-Merging, and DARE-Merging. Task Arithmetic (Ilharco et al., 2023b) enhances the merging process by introducing task vectors, suggesting that simple arithmetic operations on these vectors can effectively edit models and produce a merged model. Building on the concept of task vectors, both DARE (Yu et al., 2024b) and Ties (Yadav et al., 2023) employ pruning-then-scaling methods to merge task vectors, based on the assumption that not all parameters contribute equally to the final performance. We refer readers to Appendix 7.1 for detailed hyperparameter settings for the merging methods.

Models & Dataets Our experimental evaluation focuses on three model families: LLaMA-2 7B (Touvron et al., 2023), Mistral 7B (Jiang et al., 2023), and LLaMA-2 13B (Touvron et al., 2023), each covering distinct specializations in: general knowledge (Chat), mathematical reasoning (Math), and code generation (Code). Additional details regarding the model variants and their sources are provided in Appendix 7.2.

We assess performance using seven benchmark datasets across three domains: MMLU (Hendrycks et al., 2020), HellaSwag (Zellers et al., 2019) and TruthfulQA (Lin et al., 2022) for assessing general knowledge and reasoning capabilities; GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) for testing mathematical reasoning proficiency; HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) for evaluating code generation ability. To ensure consistent and unbiased assessment, model performance is evaluated using zero-shot accuracy, with pass@1 rate specifically measuring code generation correctness.

Calibration Data For the calibration process, we strategically selected a compact set of domain-specific data: 10 random samples per model from established datasets (GSM8K (Cobbe et al., 2021) for Math models, MBPP (Austin et al., 2021) for Code models, and Stanford Alpaca (Taori et al., 2023) for Chat models). This deliberate small-scale data sampling significantly enhances computational efficiency while maintaining effectiveness. Importantly, our analysis confirms that even with this limited calibration data, we observe stable and distinguishable divergence patterns in loss measurements across different models, providing reliable signals for sensitivity score computation.

4.1 Main Results

Merging Models with Sense-Merging. We evaluate the effectiveness of Sens-Merging by applying it as a plug-and-play module to enhance existing task-vector-based baselines. As shown in Table 1, Sens-Merging consistently improves average performance across all domains when merging Chat, Math, and Code variants of the LLaMA2-7B model. Across all methods, integrating Sens-Merging leads to notable performance gains.

(1) Significant Improvement in Task Arithmetic: Sens-Merging brings a substantial boost to Task Arithmetic, raising the average score from 29.03 to 34.78 — a 19.22% relative improvement (5.58 points). In contrast, Ties-Merging and DARE, which already apply parameter pruning to reduce interference, see more moderate gains of 0.4 and 0.27 points respectively, yet still benefit from Sens-Merging. **(2) Domain-Specific Enhancements:** In general knowledge, Sens-Merging consistently improves performance on MMLU and HellaSwag across all methods. For mathematical reasoning, the combination of Ties-Merging and Sens-Merging achieves the highest GSM8K (47.69) and MATH (7.80) scores. In code generation, Task

Method	Use Sens	General Knowledge			Mathemetical Reasoning		Code Generation		Average
		MMLU	HellaSwag	TruthfulQA	GSM8K	MATH	MBPP	HumanEval	
Chat		46.38	57.79	45.17	23.43	4.86	0.3	0.6	25.50
Math	\	40.05	56.30	32.56	48.60	8.50	21.8	12.8	31.52
Code		40.76	57.87	33.17	7.13	3.62	26.8	5.5	24.98
Task Arithmetic	✗	41.50	49.63	37.45	47.34	6.46	13.5	7.3	29.03
	✓	46.12	**59.10**	36.84	42.29	7.12	**33.1**	18.9	34.78
Ties-Merging	✗	45.75	56.63	39.89	46.93	7.74	29.1	17.1	34.73
	✓	46.03	56.87	**40.02**	**47.69**	**7.80**	29.8	17.7	**35.13**
DARE	✗	46.78	57.57	38.19	44.05	6.96	31.6	18.9	34.86
	✓	**46.81**	58.24	37.33	44.73	6.98	32.3	**19.5**	**35.13**

Table 1: Performance evaluation of merged LLaMA2-7B Models (Chat, Math, Code) across 7 task-specific datasets

Method	Use Sens	General Knowledge			Mathemetical Reasoning		Code Generation		Average
		MMLU	HellaSwag	TruthfulQA	GSM8K	MATH	MBPP	HumanEval	
Chat		59.05	65.97	55.69	42.53	9.16	49.6	42.7	46.37
Math	\	60.77	58.68	44.68	63.38	22.74	38.1	23.8	44.59
Code		50.58	53.19	45.29	31.69	4.84	50.9	40.9	39.63
Task Arithmetic	✗	47.34	46.80	41.00	52.16	13.26	32.1	29.9	37.51
	✓	**62.43**	**61.94**	45.29	**59.74**	**17.06**	54.4	34.1	47.85
Ties-Merging	✗	57.20	57.59	**48.71**	55.50	15.00	48.4	40.2	46.09
	✓	57.36	57.94	48.12	56.25	15.56	49.9	41.5	46.66
DARE	✗	55.36	55.77	42.84	57.39	15.00	49.4	39.0	44.97
	✓	58.22	58.92	46.88	58.45	16.46	**55.1**	**43.3**	**48.19**

Table 2: Performance evaluation of merged Mistral 7B Models (Chat, Math, Code) across 7 task-specific datasets

Arithmetic with Sens-Merging significantly improves MBPP (13.5 to 33.1) and HumanEval (7.3 to 18.9). **(3) Outperforming Individual Models:** Sens-Merging enables merged models to surpass their individual fine-tuned counterparts, particularly in code generation. For instance, merging Chat, Math, and Code models increases MBPP accuracy from 26.8 to 32.3 and HumanEval from 12.8 to 19.5. This shows that model merging can effectively integrate specialized capabilities, even outperforming direct fine-tuning on complex tasks.

The smaller gains observed on TIES/DARE compared to the larger improvements on Task Arithmetic stem from differences in parameter space. Task Arithmetic operates on the full model, allowing our sensitivity score to fully exploit its discriminative potential. In contrast, TIES/DARE prune 50–80% of the parameters before merging, leaving fewer parameters for Sens-Merging to optimize. As a result, many potentially valuable parameters are removed early, limiting the effectiveness of our method in these settings.

Using Different Model Architecture. To verify the generalizability of our method across architectures, we conduct experiments using Mistral-7B models. Our method demonstrates consistent performance improvements despite the architectural differences from LLaMA-based models. As shown in Table 2, when combined with Task Arithmetic and DARE, Sens-Merging demonstrated remarkable performance gains, surpassing the original baselines by 10.34 and 3.22 points respectively across all evaluated datasets. With Task Arithmetic, our method shows impressive gains across domains: 11.58 points in general knowledge, 4.86 points in mathematical reasoning, and 8.45 points in code generation. When combined with DARE, Sens-Merging particularly excelled in code generation, achieving a 5-point improvement over the original DARE and even outperforming task-specialized fine-tuned models. This superiority is evidenced by higher scores on coding benchmarks: 55.1 versus 50.9 on MBPP and 43.3 versus 40.0 on HumanEval.

Scaling to Larger Model Size. We further evaluate the scalability of our method using the LLaMA-2 13B fine-tuned models. As presented in Table 3, our approach maintains consistent performance gains at larger scales. Sens-Merging with Task Arithmetic demonstrates particularly strong improvements, outperforming the baseline by 5.68 points across all datasets, with notably impressive gains in code generation (14.75 points). When combined with Ties-Merging, Sens-Merging excels in mathematical reasoning tasks. Specifically, it achieves a 3.77% relative improvement (1.98

Method	Use Sens	General Knowledge			Mathemetical Reasoning		Code Generation		Average
		MMLU	HellaSwag	TruthfulQA	GSM8K	MATH	MBPP	HumanEval	
Chat		53.17	60.73	40.88	32.37	6.70	16.5	7.9	31.18
Math	\	52.73	61.10	37.09	55.50	10.84	28.8	15.9	37.42
Code		52.65	60.42	40.64	27.29	5.74	21.3	10.4	31.21
Task Arithmetic	✗	52.22	57.52	**41.49**	49.89	7.32	24.1	9.1	34.52
	✓	**55.88**	**61.84**	39.05	53.07	8.84	**42.6**	20.1	40.20
Ties-Merging	✗	55.48	60.65	39.05	52.46	9.90	40.4	**21.3**	39.89
	✓	55.20	60.64	39.17	54.44	**10.20**	41.3	20.6	40.22
DARE	✗	55.43	61.51	40.51	55.19	9.08	39.1	20.1	40.13
	✓	55.65	61.66	40.64	**55.42**	9.08	39.3	20.7	**40.35**

Table 3: Performance evaluation of merged LLaMA2-13B Models (Chat, Math, Code) across 7 task-specific datasets

Method	General Knowledge			Mathemetical Reasoning		Code Generation		Average
	MMLU	HellaSwag	TruthfulQA	GSM8K	MATH	MBPP	HumanEval	
Task Arithmetic	41.50	49.63	37.45	47.34	6.46	13.5	7.3	29.03
+ task-specific	41.57	49.60	**37.94**	**48.29**	**7.84**	13.3	7.3	29.41
+ cross-task	45.99	59.07	36.35	42.00	7.00	32.1	18.3	33.40
+ Sens-Merging	**46.12**	**59.10**	36.84	42.29	7.12	**33.1**	**18.9**	**34.78**
Ties-Merging	45.75	56.63	39.89	46.93	7.74	29.1	17.1	34.73
+ task-specific	**47.34**	56.63	38.68	46.63	7.92	29.3	17.7	34.89
+ cross-task	45.20	**57.10**	**40.17**	**47.89**	7.32	29.6	17.3	34.94
+ Sens-Merging	46.03	56.87	40.02	47.69	**7.80**	**29.8**	**17.7**	**35.13**

Table 4: Ablation studies on task-specific scaling and cross-task scaling for Task Arithmetic in LLaMA2 7B models.

points) on the GSM8K dataset and a 3.03% relative improvement on the MATH dataset.

4.2 Ablation Studies

To evaluate each component, we perform ablation studies using task-specific and cross-task scaling within Task Arithmetic and Ties-Merging. As shown in Table 4, Task Arithmetic shows task-dependent behavior. Task-specific scaling improves MATH by 1.38 points (21.36% relative gain) but has limited impact elsewhere. In contrast, cross-task scaling boosts general knowledge and code generation by 4.28 and 14.8 points, respectively, though it reduces math accuracy by 4.8 points. Overall, cross-task scaling achieves the highest average gain of 5.37 points. For Ties-Merging, similar trends emerge. Task-specific scaling yields the best MMLU score (47.34), highlighting its strength in factual knowledge retention. Cross-task scaling improves TruthfulQA and GSM8K, indicating better generalization, but offers smaller gains on specialized tasks. The best overall performance (35.13) is achieved when Ties-Merging is combined with Sens-Merging, balancing performance across all domains.

Therefore, each scaling method involves a trade-off: task-specific scaling excels at enhancing specialized capabilities (particularly mathematical reasoning) but with limited broader impact, while cross-task scaling offers stronger overall performance improvements at the cost of sacrificing some task-specific excellence.

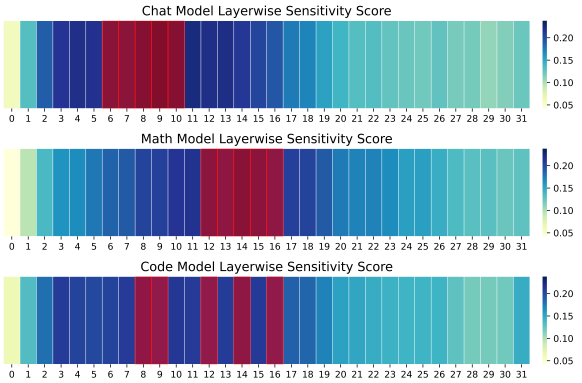

Figure 3: Layer-wise sensitivity scores across different task-specific models, with the Top-5 most sensitive layers highlighted in red.

5 In-depth Analysis

5.1 Scaling Factors Analysis

Layerwise Sensitivity Distribution. Figure 3 reveals distinct layer-wise sensitivity patterns across model specializations: the Chat model peaks at layer 10, leveraging lower layers for language processing; the Math model shows maximum sensitivity around layer 15, emphasizing middle layers for mathematical reasoning; and the Code model exhibits a unique dual-peak pattern, reflecting its

Merging Methods	Models	Use Sens	General Knowledge			Mathematical Reasoning		Code Generation		Average
			MMLU	HellaSwag	TQA	GSM8K	MATH	MBPP	HumanEval	
/	Chat	/	46.38	57.79	45.17	23.43	4.86	0.3	0.6	25.5
	Math		40.05	56.30	32.56	48.60	8.50	21.8	12.8	31.52
	Code		40.76	57.87	33.17	7.13	3.62	26.8	5.5	24.98
Task Arithmetic	Chat &Math	✗	41.36	49.77	36.96	45.34	6.96	13.8	7.3	28.78
		✓	**45.67**	58.02	**37.21**	45.49	7.14	**30.1**	**17.7**	**34.48**
	Math &Code	✗	40.54	56.63	32.80	**50.42**	**9.38**	22.6	10.4	31.82
		✓	43.67	**59.15**	33.90	42.08	7.12	27.8	16.5	32.89
Ties-Merging	Chat &Math	✗	45.84	56.48	39.17	48.29	7.86	30.1	17.1	34.98
		✓	45.86	56.63	**40.02**	**48.98**	7.92	**31.3**	**17.7**	**35.49**
	Math&Code	✗	42.52	58.30	36.72	45.11	8.44	29.1	14.6	33.54
		✓	42.74	**58.55**	36.96	45.49	**8.54**	30.3	14.8	33.91
DARE	Chat&Math	✗	46.72	58.04	**39.53**	44.88	6.58	30.3	20.1	35.16
		✓	**46.78**	58.12	**39.53**	**45.03**	**7.06**	31.6	**20.7**	**35.55**
	Math&Code	✗	43.95	59.21	33.78	39.80	6.34	29.6	17.1	32.83
		✓	43.98	**59.25**	33.90	40.41	6.82	29.8	17.1	33.04

Table 5: Performance evaluation of two merged fine-tuned LLaMA2-7B models (Chat&Math and Math&Code) across seven task-specific datasets.

Stage	Task-specific Coefficients	Cross-task Coefficients	Ties (80% drop)	Ties (disjoint merging)
Time Required	2 min 31 sec	1 min 53 sec	3 min 4 sec	1 min 13 sec

Table 6: Computational time (in minutes:seconds) of merging components in Sens-Merging and TIES-Merging, demonstrating the lightweight nature of our approach on LLaMA2 7B Math model.

need for both linguistic processing in lower layers and logical reasoning in middle layers. Thus, by leveraging layer-wise sensitivity, we enhance the weights of the layers that are most critical to performance, thereby ensuring that specialized functions are optimally preserved.

Models	LLaMA2 7B	Mistral 7B	LLaMA2 13B
Chat	0.3095	0.2454	0.2958
Math	**0.5062**	**0.6379**	**0.5284**
Code	0.1843	0.1167	0.1758

Table 7: Cross-task scaling coefficients across Chat, Math, and Code models.

Cross-Task Sensitivity Scaling. In Table 7, we observe consistent variations in cross-task scaling factors across task vectors. Math models show the highest scaling coefficients (0.51-0.64), followed by Chat models (0.25-0.31), and Code models (0.12-0.18). These coefficients reveal that: mathematical reasoning provides strong transferable skills across tasks, chat abilities facilitate general language understanding, while coding skills are more specialized and less transferable.

5.2 Merge Two Fine-tuned Models

We also evaluate Sens-Merging on two-model combinations: Chat & Math and Math & Code. We omit Chat & Code due to its significantly lower performance compared to the three-model setup. As shown in Table 5, Sens-Merging outperforms baselines in two-model settings, delivering a 5.70-point gain in Task Arithmetic. On code generation, it improves MBPP performance by 3.99% (1.2 points) over Ties-Merging and 4.29% (1.3 points) over DARE. Interestingly, for both Ties-Merging and DARE, merging only Chat and Math models outperforms the full three-model combination, achieving higher average scores (34.98 vs. 34.73 and 35.16 vs. 34.86, respectively). This suggests that the Math model may already capture much of the Code model's capabilities, and reducing model count helps minimize parameter interference, leading to better overall performance.

5.3 Computational Resource Consumption

To evaluate the efficiency of our Sens-Merging framework, we measure the computational overhead involved in computing task-specific and cross-task scaling coefficients, as well as the weight dropping and merging stages in the Ties method. All measurements are performed on CPU core. As summarized in Table 6, for the LLaMA2 7B Math model, determining the task-specific coefficients requires 2 minutes and 31 seconds, while computing the cross-task coefficients takes 1 minute and

Calibration	MMLU	HellaSwag	TruthfulQA	Math	GSM8K	MBPP	HumanEval	Average
Task Arithmetic (10 samples)	46.12	**59.10**	36.84	42.29	**7.12**	**33.1**	**18.9**	**34.78**
Task Arithmetic (20 samples)	**46.63**	58.76	**37.34**	**42.98**	6.88	**33.1**	17.7	34.77

Table 8: Performance comparison of Task Arithmetic with different calibration sample sizes.

Method	MMLU	HellaSwag	TruthfulQA	Math	GSM8K	MBPP	HumanEval	Average
Task Arithmetic	47.34	46.80	41.00	52.16	13.26	32.1	29.9	37.51
WIDEN	49.97	50.12	**46.14**	48.45	12.30	41.6	**43.3**	41.70
Sens-Merging (TA)	**62.43**	**61.94**	45.29	**59.74**	17.06	**54.4**	34.1	**47.85**
TIES	57.20	57.59	48.71	55.50	15.00	48.4	40.2	46.09
DARE	55.36	55.77	42.84	57.39	15.00	49.4	39.0	44.97

Table 9: Performance comparison between WIDEN and Sens-Merging on merging Mistral-7B models.

53 seconds. The task-specific calculation is slightly slower due to gradient information and backward propagation, compared to the forward pass needed for the logits-based cross-task estimation. Despite this, the time taken for the TIES method to drop 80% of the model's weights is 3 minutes and 4 seconds and another 1 minute and 13 seconds for disjoint merging. The lightweight nature of our calibration process ensures that Sens-Merging remains practical and accessible, requiring only marginal computational resources beyond those needed for standard model merging approaches, while delivering superior performance across the evaluation metrics.

5.4 Impact of Calibration Data

To evaluate the sensitivity of our method to calibration data size, we conducted experiments using both 10 and 20 calibration samples per model. As shown in Table 8, increasing the calibration data size from 10 to 20 samples results in minimal performance variation across all tasks.

The results indicate that while calibration data selection is an important consideration, our method remains robust even with a small number of samples. The stable and distinguishable differences in loss and logits across models provide sufficient signal for effective sensitivity estimation. This confirms the practical efficiency of our approach and supports the use of lightweight calibration without significant compromise in performance.

5.5 Comparison with WIDEN

We further compare our Sens-Merging framework with WIDEN (Yu et al., 2024a), a method that identifies important parameters using magnitude and directional criteria. While WIDEN performs well on models with large weight divergence (e.g., pre-trained and fine-tuned checkpoints), it shows reduced effectiveness when merging closely related task-specific models such as Mistral-7B variants. As shown in Table 9, WIDEN improves general knowledge but falls short on math tasks, revealing limitations in preserving task-specific performance across models. In contrast, our Sens-Merging consistently enhances performance across all domains.

Importantly, Sens-Merging is designed as a plug-and-play module that seamlessly integrates with existing model merging frameworks such as TIES and DARE, further boosting their performance. This flexibility enables users to adopt Sens-Merging without extensive architectural modifications. Finally, unlike WIDEN, which relies solely on parameter statistics, Sens-Merging leverages activation-based calibration data to guide the merging process. This allows precise control over task-specific behaviors, particularly in long-to-short (L2S) reasoning scenarios (Wu et al., 2025).

6 Conclusion

We introduce Sens-Merging, a novel method that determines model merging coefficients by analyzing parameter sensitivity both within specific tasks and across multiple tasks. Through extensive evaluation on Mistral 7B and LLaMA-2 7B/13B model families, we demonstrate that Sens-Merging enhances model merging performance across multiple domains, consistently outperforming both existing merging techniques and individually fine-tuned models. This improvement is particularly pronounced in code generation tasks, where merged models achieve superior results compared to specialized fine-tuning.

Limitations

While Sens-Merging demonstrates remarkable performance in model merging, achieving consistent improvements across various benchmarks, it shares fundamental limitations with existing task arithmetic-based methods. For example, our current implementation primarily addresses homogeneous model merging where base models share identical architectures. While this focus allows us to achieve state-of-the-art results in such scenarios, extending Sens-Merging to heterogeneous architectures remains an exciting direction for future research. Moreover, our method is specifically designed for large language models and has been primarily validated with LoRA fine-tuned models, where weight differences between specialized models are relatively constrained. For smaller-scale models or fully fine-tuned models with larger weight divergences, our approach may require adaptations.

Ethics Statement

This study utilizes publicly available datasets for our models. Prior research endeavors have generally taken ethical considerations into account. We have manually inspected a subset of samples and found no explicit ethical concerns, including violent or offensive content. Nonetheless, it is crucial to highlight that the output generated by large language models lacks the degree of control we might assume. Consequently, we are prepared to implement measures to mitigate any unforeseen outputs.

References

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha S. Srinivasa. 2023. Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Anton Alexandrov, Veselin Raychev, Mark Mueller, Ce Zhang, Martin T. Vechev, and Kristina Toutanova. 2024. Mitigating catastrophic forgetting in language transfer via model merging. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 17167–17186. Association for Computational Linguistics.

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

Omri Avrahami, Dani Lischinski, and Ohad Fried. 2022. GAN cocktail: Mixing gans without dataset access. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIII*, volume 13683 of *Lecture Notes in Computer Science*, pages 205–221. Springer.

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

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

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. 2022. The role of permutation invariance in linear mode connectivity of neural networks. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34:27503–27516.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. 2020. Linear mode connectivity and the lottery ticket hypothesis. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3259–3269. PMLR.

C. Daniel Freeman and Joan Bruna. 2017. Topology and geometry of half-rectified network optimization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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: EMNLP 2024 - Industry Track, Miami, Florida, USA, November 12-16, 2024*, pages 477–485. Association for Computational Linguistics.

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

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*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023a. Editing models with task arithmetic. *Preprint*, arXiv:2212.04089.

Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023b. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Keller Jordan, Hanie Sedghi, Olga Saukh, Rahim Entezari, and Behnam Neyshabur. 2023. REPAIR: renormalizing permuted activations for interpolation repair. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252.

Michael Matena and Colin Raffel. 2022. Merging models with fisher-weighted averaging. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Dang Nguyen, Trang Nguyen, Khai Nguyen, Dinh Q. Phung, Hung Hai Bui, and Nhat Ho. 2023. On cross-layer alignment for model fusion of heterogeneous neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*, pages 1–5. IEEE.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Ken Shoemake. 1985. Animating rotation with quaternion curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1985, San Francisco, California, USA, July 22-26, 1985*, pages 245–254. ACM.

George Stoica, Daniel Bolya, Jakob Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. 2024. Zipit! merging models from different tasks without training. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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`.

N. Joseph Tatro, Pin-Yu Chen, Payel Das, Igor Melnyk, Prasanna Sattigeri, and Rongjie Lai. 2020. Optimizing mode connectivity via neuron alignment. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen, Tao Zhong, and Mingxuan Yuan. 2025. Unlocking efficient long-to-short llm reasoning with model merging. *arXiv preprint arXiv:2503.20641*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

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. *CoRR*, abs/2408.07666.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024a. Extend model merging from fine-tuned to pre-trained large language models via weight disentanglement. *CoRR*, abs/2408.03092.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024b. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

7 Appendix

7.1 Hyperparameters

Both baselines and our Sens-Merging enhanced baselines use the same hyperparameters for fair comparison. For Task Arithmetic, we use a default scaling coefficient of $\lambda = 1$, which maintains the original magnitude of task vector when adding the pretrained backbone. However, the DARE method has been observed to be more sensitive to variations in both the scaling coefficient λ and the drop rate parameter r. To achieve a balanced performance, we set the scaling coefficient to $\lambda = 0.5$ and establish a default drop rate of $r = 0.5$ for DARE. Similarly, for Ties-Merging, which requires the specification of a masking ratio, we set the default mask ratio to $r = 0.7$ across all experiments.

7.2 Model Variants and Sources

We evaluate our method across multiple fine-tuned variants of LLaMA2-7B, Mistral 7B, and LLaMA2-13B models, focusing on specialized capabilities in instruction-following (Chat), mathematical reasoning (Math), and code generation (Code). Table 10 summarizes the fine-tuned variants used for LLaMA2-7B along with their corresponding Hugging Face identifiers and use cases. For completeness, we also include similarly fine-tuned versions from other base models: Mistral 7B (Table 11) and LLaMA2-13B Table 12. These variants are obtained from publicly available checkpoints and are used consistently across all compression and merging experiments reported in this work.

	Model Type	Hugging Face Identifier	Use Case
LLaMA2-7B	Pretrained Base Model	meta-llama/Llama-2-7b-hf	General language modeling
	Chat-Tuned Model	meta-llama/Llama-2-7b-chat-hf	Dialogue and instruction-following
	Math-Tuned Model	TIGER-Lab/MAmmoTH-7B	Mathematical reasoning
	Code-Tuned Model	mrm8488/llama-2-coder-7b	Code generation

Table 10: Fine-tuned variants and corresponding Hugging Face identifiers for LLaMA-2 7B models.

	Model Type	Hugging Face Identifier	Use Case
Mistral 7B	Pretrained Base Model	mistralai/Mistral-7B-v0.1	General language modeling
	Chat-Tuned Model	mistralai/Mistral-7B-Instruct-v0.1	Dialogue and instruction-following
	Math-Tuned Model	TIGER-Lab/MAmmoTH2-7B	Mathematical reasoning
	Code-Tuned Model	Nondzu/Mistral-7B-codealpaca-lora	Code generation

Table 11: Fine-tuned variants and corresponding Hugging Face identifiers for Mistral 7B models.

	Model Type	Hugging Face Identifier	Use Case
LLaMA2-13B	Pretrained Base Model	meta-llama/Llama-2-13b-hf	General language modeling
	Chat-Tuned Model	meta-llama/Llama-2-13b-hf	Dialogue and instruction-following
	Math-Tuned Model	TIGER-Lab/MAmmoTH-13B	Mathematical reasoning
	Code-Tuned Model	emre/llama-2-13b-code-chat	Code generation

Table 12: Fine-tuned variants and corresponding Hugging Face identifiers for LLaMA-2 13B models.