
Decouple and Orthogonalize: A Data-Free Framework for LoRA Merging

**Shenghe Zheng^{1,2}, Hongzhi Wang¹, Chenyu Huang³, Xiaohui Wang³, Tao Chen³,
Jiayuan Fan³, Shuyue Hu², Peng Ye^{2,4}**

¹ Harbin Institute of Technology ² Shanghai AI Laboratory ³ Fudan University

⁴ The Chinese University of Hong Kong

shenghe.zheng@gmail.com

Abstract

With more open-source models available for diverse tasks, model merging has gained attention by combining models into one, reducing training, storage, and inference costs. Current research mainly focuses on model merging for full fine-tuning, overlooking the popular LoRA. However, our empirical analysis reveals that: a) existing merging methods designed for full fine-tuning perform poorly on LoRA; b) LoRA modules show much larger parameter magnitude variance than full fine-tuned weights; c) greater parameter magnitude variance correlates with worse merging performance. Considering that large magnitude variances cause deviations in the distribution of the merged parameters, resulting in information loss and performance degradation, we propose a **Decoupled and Orthogonal** merging approach (**DO-Merging**). By separating parameters into magnitude and direction components and merging them independently, we reduce the impact of magnitude differences on the directional alignment of the merged models, thereby preserving task information. Furthermore, we introduce a data-free, layer-wise gradient descent method with orthogonal constraints to mitigate interference during the merging of direction components. We provide theoretical guarantees for both the decoupling and orthogonal components. And we validate through extensive experiments across vision, language, and multi-modal domains that our proposed DO-Merging can achieve significantly higher performance than existing merging methods at a minimal cost. Notably, each component can be flexibly integrated with existing methods, offering near free-lunch improvements across tasks.

1 Introduction

Deep learning is widely used in many applications [5, 35]. However, edge-side users often lack strong resources or large datasets, and thus prefer ready-to-use models tailored to their specific tasks. The rapid growth of open-source platforms like HuggingFace [37] has made this goal increasingly achievable. In practice, real-world tasks often involve multiple subtasks [43, 19]. Handling each with a separate model increases cost and deployment complexity. Model merging addresses this by combining existing models into a single model capable of handling all target tasks, reducing both retraining and deployment costs [18, 26, 41]. This approach has recently gained great attention.

Current research on model merging mainly targets task interference. Methods are categorized by their target models: full fine-tuning or PEFT techniques like LoRA [14]. For full fine-tuning, merging approaches fall into three types: automatic coefficient computation [43, 19, 26], optimization-based conflict reduction using task vectors [41, 4, 8, 53], and dedicated modules for task-specific knowledge [17, 25]. However, these methods may not be suitable for LoRA, or require architectural changes, limiting usability. Thus, specialized LoRA merging techniques are needed. Existing

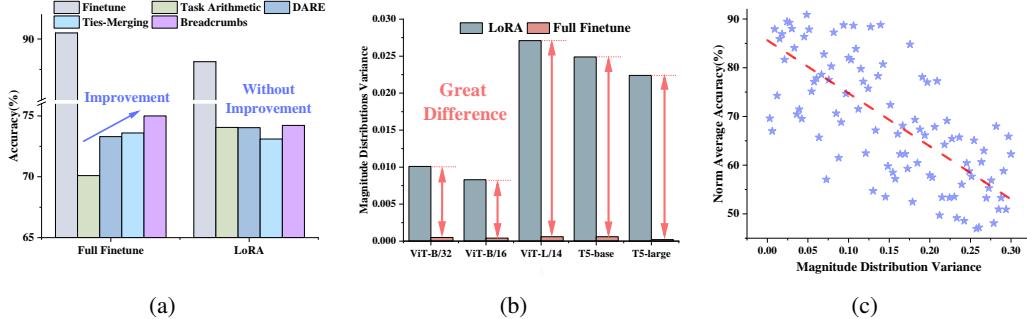


Figure 1: Key Observations on LoRA Merging. (a) Existing methods work well for full fine-tuning but fail on LoRA. (b) LoRA shows larger parameter discrepancies across tasks than full fine-tuning. The Magnitude Distribution Variance is calculated as discussed in Appendix C.4. (c) A greater parameter discrepancy between models correlates with worse merging performance.

LoRA-specific methods mainly include subspace projection merging [33, 52] and parameter-driven coefficient computation [2, 27, 29, 16]. Yet, they often result in limited gains or lack generalization across tasks. Therefore, new and more effective merging strategies are necessary.

We begin by analyzing why merging strategies designed for full fine-tuning perform poorly on LoRA as shown in Fig. 1(a). We find that LoRA modules trained on different tasks show larger parameter distribution ranges compared to full fine-tuning as shown in Fig. 1(b), and demonstrate in Fig. 1(c) that the performance degradation is closely related to the distribution discrepancies. We propose that this loss stems from the merging process aims to preserve both the distribution scale (magnitude) and shape (direction) of parameters to maintain input mapping patterns [45]. However, in LoRA, modules with larger magnitudes dominate the parameter distribution of merged result, leading to shifts in the range and shape of the merged distribution, which causes partial task information loss and hence performance degradation on certain tasks.

To reduce the impact of magnitude on merging, we propose to decouple magnitude and direction, allowing the direction merging to be free from magnitude interference. During direction merging, we further aim to minimize the interference between different tasks in low cost. Then, we introduce **DO-Merging** (Decouple and Orthogonalize) framework as shown in Fig. 2. We first decompose each parameter matrix into a magnitude and direction vector to merge separately, using the normalization coefficient of each column as magnitude and the remaining part as direction. This decoupling ensures that merging is not biased by magnitude differences. As for the magnitude vectors, to retain the scale distributions of each task, we perform average merging. For the direction vectors, we apply data-free, layer-wise gradient descent to generate orthogonal perturbations, minimizing mutual task interference while preserving task-specific information. Since LoRA merging is performed in the full-rank space as discussed in D.4, we apply orthogonalization to both LoRA components separately before decoupling in the full-rank matrix for lower cost. This does not affect the conclusion. We then combine the two components to form the final merged model.

We provide theoretical analysis to support DO-Merging’s effectiveness. Experiments across vision, language, and multi-modal tasks confirm its broad applicability. Notably, it achieves over 3% improvement at minimal cost on various vision tasks, and consistent gains on large language and multi-modal models. And both components can be integrated with existing methods, offering up to nearly 4% free-lunch improvements across multiple tasks. The main contributions are as follows:

- We empirically demonstrate that existing model merging methods for full fine-tuning perform poorly on LoRA. We show that it is closely related to the greater distribution variations exhibited by LoRA compared to full fine-tuning, as supported by both experiments and theory.
- We propose DO-Merging, the first method addressing LoRA merging degradation via a decoupled and orthogonal perspective. We decouple parameters into magnitude and direction, merging them separately to mitigate magnitude effects. We further apply a data-free orthogonal constraint to reduce direction interference. Both parts come with theoretical guarantees.
- We validate the effectiveness of our DO-Merging through comprehensive experiments covering vision, language, and multi-modal tasks. Moreover, both components of DO-Merging can be flexibly combined with current approaches, bringing more than a 3% performance gain.

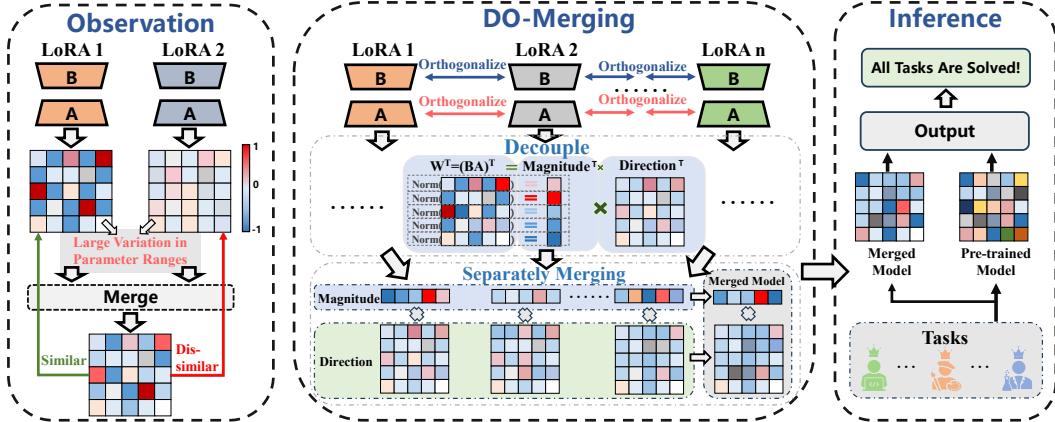


Figure 2: DO-Merging Framework. Left: Large magnitude differences in LoRA across tasks degrade merging performance. Middle: DO-Merging process—orthogonal perturbation, decoupling magnitude and direction, and separate merging. Right: Single model deployment for multiple tasks.

2 Related Work

Model Merging. Model Merging combines existing models trained on different tasks into a single model that can handle multiple tasks without further training [18, 19, 42]. The key challenge is resolving task conflicts during merging [41, 30, 54]. Existing methods for fully fine-tuned models fall into three categories: automatic computation of merging coefficients [43, 19, 26], parameter-based conflict reduction [18, 43, 8, 10, 30, 11], or storing additional knowledge to mitigate interference [17, 25]. However, these approaches either require extensive data or computation, rely on task-specific designs, or demand major architectural changes, limiting their use by general users. For more details, please refer to Appendix C.2. When applied to LoRA, these methods suffer from performance drops due to large magnitude differences among modules. To address this, we propose DO-Merging.

LoRA Merging. Current LoRA merging methods fall into two main categories: automatic merging coefficient computation [2, 32, 16, 29, 47, 44] and parameter adjustment to reduce task conflicts [52, 33]. The former often requires data or heavy computation, limiting its accessibility for general users. The latter relies on task-specific constraints and custom designs, reducing usability. Additionally, model swarm-style methods require data and suffer from high inference costs, making them impractical for general users. Details can be found in Appendix C.2. We find that the key challenge in LoRA merging stems from large magnitude differences among LoRAs. To address this, we propose DO-Merging.

Weight Decouple. Weight Decouple in deep learning is typically used during training, such as applying distinct objectives for each component [22, 15, 48]. Decoupling with fixed weights is often used to compute merging coefficients [45, 40]. Our work is the first to apply decoupling and orthogonalization in a no-retraining setting for reducing task conflicts during merging.

3 Methodology

3.1 Preliminary

Given a base model θ_{pre} , let $\{\Delta_i\}_{i=1}^n$ denote the LoRA modules for n tasks, each containing K layers. The k -th layer of Δ_i is denoted as a matrix $W_i^k = B_i^k A_i^k$, where B_i^k and A_i^k are the corresponding LoRA parameters. We aim to construct a merged module $\Delta = \lambda \sum_{i=1}^n g(\Delta_i)$ such that the model $\theta_{pre} + \Delta$ can solve all n tasks simultaneously, where g denotes a transformation function.

3.2 Motivation

As shown in Fig. 1(b), LoRA modules exhibit larger magnitude differences during training than full fine-tuning, and these differences are closely related to the drop in merging performance. Based on this observation, we propose that the merging process should preserve both the distribution shape

(direction) and magnitude of the parameters, in order to maintain approximately the same input mapping. However, when the parameter distributions of the models to be merged vary significantly in scale, the directions of the modules with larger ranges tend to dominate, leading to loss of directional information from other modules and thus performance degradation. We provide theoretical support for the relationship between magnitude differences and merging performance. Proof is in Appendix B.1.

Assumption 3.1. Consider two matrices $W_1, W_2 \in \mathbb{R}^{m \times n}$, and assume $W_i = \alpha_i \times \bar{W}_i$, where $\alpha_i \in \mathbb{R}^{1 \times n}$ is the magnitude and $\bar{W}_i[:, j] \sim \mathcal{N}(0, 1)$. We assume that the merged matrix W preserves features when close to the original matrices, with performance negatively correlated with the loss:

$$L = \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_1\|_2} \|W - W_1\|_2 + \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_2\|_2} \|W - W_2\|_2. \quad (1)$$

Theorem 3.1. $\mathbb{E}(L)$ achieves its minimum when $\|\alpha_1\|_2 = \|\alpha_2\|_2$, and is greater than this minimum in both cases $\|\alpha_1\|_2 > \|\alpha_2\|_2$ and $\|\alpha_1\|_2 < \|\alpha_2\|_2$.

Theorem 3.1 theoretically supports the observation in Fig. 1(c) that large magnitude differences degrade merging performance. We propose that since magnitude is the key factor causing performance degradation, we decouple magnitude and direction in the merging process. The benefit of this decoupling is to reduce directional information loss caused by magnitude differences during merging. For the magnitude vectors, to preserve the magnitude distribution across models, we merge different magnitude vectors, as detailed in Sec. 3.3. To reduce merging loss for direction vectors, we propose an efficient, data-free orthogonalization method, also described in Sec. 3.4. Combining these two components yields the final merged model in Sec. 3.5. Since LoRA involves two low-rank matrix components, we perform orthogonalization on the two low-rank matrices first for reducing cost, and then decouple the full-rank matrices. Although the order is changed, conclusions remain unaffected.

3.3 Decouple

First, we introduce a decoupling method to isolate magnitude differences into magnitude vectors, leaving direction vectors with consistent magnitudes. We extract column normalization coefficients as magnitude vectors. For a task vector layer $W = BA \in \mathbb{R}^{m \times n}$, the decoupling process is:

$$W = \alpha \bar{W}, \quad \alpha_{[1,j]} = \text{norm}(W_{[:,j]}). \quad (2)$$

Here, $\alpha \in \mathbb{R}^{1 \times n}$ represents the magnitude vector, and \bar{W} represents the direction vector. The norm function denotes column normalization of matrix.

Column normalization is chosen as the magnitude vector because for a neural network layer with pre-trained parameters W_{pre} and task-specific parameters W_t , the input x and output x^{out} satisfy:

$$x^{out} = x(W_{pre} + W_t) = xW_{pre} + xW_t = xW_{pre} + \omega, \quad (3)$$

where $\omega_{[i,j]} = \sum_k x_{[i,k]}(W_t)_{[k:j]}$. This indicates that each column of the task vector contributes jointly to the output. Thus, aligning column magnitudes across different task vectors helps preserve their individual output characteristics.

For magnitude vectors, to retain amplitude features from different models after merging, we perform average merging on the magnitude vectors. Assuming we have n task vectors, we define the merged magnitude vector at each layer as: $\alpha_{merge} = \sum_{i=1}^n \alpha_i$. Without optimizing direction vectors, we provide theoretical guarantees by the following theorem that our proposed decoupling method improves merging performance. Detailed proof can be found in Appendix B.2.

Theorem 3.2. Let the merging parameters be W_1 and W_2 . The non-decoupled merged model is $W^1 = \lambda_1 \sum_{i=1}^2 \alpha_i \bar{W}_i$, and the decoupled one is $W^2 = \lambda_2 \sum_{j=1}^2 \alpha_j \sum_{i=1}^2 \bar{W}_i$. When $\|\alpha_1\|_2 \neq \|\alpha_2\|_2$, the expected loss as Eq. 1 satisfies $\mathbb{E}(L_2) < \mathbb{E}(L_1)$, which implies that W^2 outperforms W^1 .

3.4 Orthogonalize

Next, we introduce the optimization of direction vectors. For different task vectors, when two task vectors are nearly orthogonal, it indicates that the corresponding features in the parameter space are also nearly orthogonal, resulting in minimal task interference during merging [10]. Previous work has shown that imposing orthogonality constraints between LoRA modules during fine-tuning

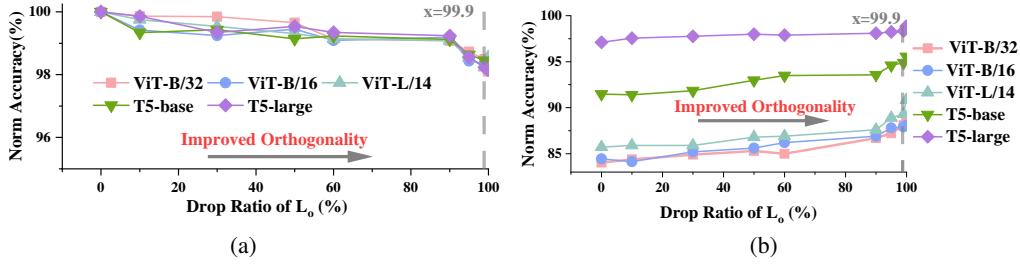


Figure 3: Key observations on orthogonalization. (a) Average Norm Performance change of task models during orthogonal gradient descent. Performance remains stable. (b) As orthogonality increases, the average merged norm accuracy also improves.

is beneficial for merging [48], but publicly available weights often do not meet this requirement. Additionally, ensuring orthogonality using data requires significant amounts of data and substantial forward and backward propagation costs. We aim to achieve similar results using a data-free approach.

Given the high redundancy in fine-tuned parameters, minor adjustments typically do not affect task performance. We therefore propose to apply orthogonality constraints directly to the parameters of existing LoRA modules without data. As illustrated in Fig. 3(a), this optimization has little negative impact on the performance of individual tasks. Nevertheless, Fig. 3(b) shows that such orthogonality without considering data factors can still significantly improve the effectiveness of model merging on various models. Therefore, we propose applying data-free and layer-wise parameter-level orthogonality constraints. Thus, for each layer of task vectors, we construct a loss function as follows:

$$L = \sum_i \sum_j (W_i + \delta_i)^T (W_j + \delta_j) + \sum_i \|\delta_i\|_2 = L_o + L_r. \quad (4)$$

Since this optimization is layer-wise, the cost is minimal. For LoRA, we apply orthogonality separately on A and B , which is equivalent to orthogonality on the product matrix, further reducing cost. We provide theoretical guarantees for the performance improvement due to this orthogonalization by the following theorem. Detailed proof can be found in Appendix B.3.

Theorem 3.3. *As $\|\delta_i\|_2 \rightarrow 0$, smaller values of $\|W_i^T W_j\|$ lead to less conflict during merging.*

3.5 Workflow

So far, we have obtained the merged results for both the direction vectors and the magnitude vectors. For each merged weight, we can compute it using the following formula:

$$W_{out} = W_{pre} + \lambda \left(\sum_{i=1}^n \alpha_i \left(\sum_{j=1}^n \overline{W}_j \right) \right). \quad (5)$$

We also provide the pseudo-code for the overall procedure, as outlined in Alg. 1. We first apply orthogonality constraints separately to the two low-rank matrices of LoRA (lines 2–3). Then, we can obtain orthogonal full-rank matrices. After decomposing magnitude and direction from these matrices, we merge them separately to obtain the final result (lines 5–10).

Algorithm 1 Workflow of DO-Merging

Input: Fine-tuned LoRAs $\{B_i, A_i\}_{i=1}^n$ for each layer, pre-trained model W_{pre} .

- 1: **Step1: Orthogonalize;**
- 2: $\{\hat{A}_i\}_{i=1}^n = Ortho\{A_i\}_{i=1}^n$ as Eq. 4.
- 3: $\{\hat{B}_i\}_{i=1}^n = Ortho\{B_i\}_{i=1}^n$ as Eq. 4.
- 4: **Step2: Decouple;**
- 5: $\{W_i\}_{i=1}^n = \{\hat{B}_i \hat{A}_i\}_{i=1}^n$
- 6: **for** $i \in [1, n]$ **do**
- 7: Get \overline{W}_i, α_i as Eq. 2.
- 8: **end for**
- 9: **Step3: Merge;**
- 10: $W_{out} = W_{pre} + \lambda \sum_{i=1}^n \alpha_i \sum_{j=1}^n \overline{W}_j$

Output: Merged model W_{out} .

4 Experiment

In this part, we conduct comprehensive experimental analyses across various tasks. From Sec. 4.1 to Sec. 4.3, we perform experiments on multiple tasks using vision models, medium language models, large language models, and multi-modal models, respectively. In Sec. 4.4, we present ablation studies about our DO-Merging, and in Sec. 4.5, we provide an in-depth discussion of our methods.

Table 1: Multi-task performance when merging ViT-B/32 on eight vision tasks.

Method	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Pre-trained	62.30	59.70	60.70	45.50	31.40	32.60	48.50	43.80	48.06
Finetune	71.81	71.23	94.33	98.85	97.19	98.62	99.63	73.40	88.13
Task Arithmetic [18]	63.38	60.27	75.46	87.70	84.73	70.74	96.64	53.55	74.06
Ties Merging [41]	63.81	53.95	73.56	86.00	88.27	71.15	97.81	50.32	73.11
Breadcrumbs [4]	64.02	57.87	74.12	88.73	84.24	74.28	97.29	53.23	74.22
AdaMerging [43]	62.59	57.89	75.94	90.63	78.93	88.82	96.92	52.07	75.47
PCB-Merging [8]	63.74	58.64	73.58	88.12	85.65	75.12	97.45	54.07	74.54
KNOTS [33]	64.08	59.26	76.11	86.93	86.94	75.54	96.85	57.46	75.39
TSVM [10]	64.81	59.31	78.73	89.63	85.43	76.63	97.02	57.93	76.19
CoPA-Merging [47]	63.40	59.56	76.68	88.93	85.54	77.00	97.14	56.56	75.60
DO-Merging(ours)	64.48	61.01	77.30	88.30	89.05	83.31	98.03	61.53	77.88

Table 3: Multi-task performance when merging T5-base on eight language tasks.

Method	COLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Pre-trained	69.1	56.5	76.2	88.4	82.1	80.1	91.2	62.2	75.7
Finetune	69.1	82.7	85.5	90.9	84.0	84.5	92.9	87.4	84.6
Task Arithmetic [18]	68.8	55.2	78.7	89.8	83.7	79.1	91.5	72.4	77.4
Ties Merging [41]	68.3	56.3	79.4	89.8	83.7	79.4	91.6	71.2	77.5
Breadcrumbs [4]	68.3	57.3	78.5	89.8	83.6	79.5	91.8	72.4	77.6
PCB-Merging [8]	68.3	57.4	79.3	89.8	83.6	79.2	91.5	72.6	77.7
KNOTS [33]	69.4	63.6	78.5	89.8	83.6	80.2	91.5	70.6	78.4
TSVM [10]	69.1	67.8	79.9	89.7	83.6	80.9	91.6	69.4	79.0
CoPA-Merging [47]	69.4	59.0	79.2	89.9	83.6	80.1	91.4	75.4	78.5
DO-Merging(ours)	69.3	75.8	79.7	90.5	84.0	80.1	93.1	74.4	80.9

4.1 Vision Models

In this part, we conduct experiments on Vision Transformers (ViT) [6] of different scales. We select ViT-B/32, ViT-B/16, and ViT-L/14, and perform LoRA fine-tuning on each model across eight visual classification tasks, followed by merging the fine-tuned result to evaluate multi-task performance. The detailed fine-tuning and merging configurations are provided in Appendix C.1. The vision tasks used are SUN397, Cars, RESISC45, EuroSAT, SVHN, GTSRB, MNIST, and DTD [18]. The experimental results can be found in Tab 1 and Tab 2. Additional results are provided in Appendix F.

By analyzing the results, we observe that our method demonstrates significant advantages over existing approaches in the merging experiments across all three ViT scales. The average performance improvement across these three ViTs is approximately 2%. Notably, our method shows substantial gains on certain tasks with all three models, indicating that task-specific parameter features of these tasks suffer from considerable task interference with model merging methods without decoupling and orthogonality. Then, with our DO-Merging, such task interference is reduced, leading to greatly improved performance of the merged model on various target tasks. This strongly demonstrates the effectiveness of our DO-Merging approach.

4.2 Language Models

In this section, we conduct experiments on language models of various scales to demonstrate the generalization capability of our proposed DO-Merging. For medium-sized models, we selected T5-base and T5-large [31] as base models and use eight discriminative language tasks as target tasks.

Table 2: Average multi-task performance when merging ViT-B/16 and ViT-L-14 on eight tasks.

Method	ViT/B-16	ViT/L-14
Pre-trained	55.20	64.89
Finetune	89.94	93.11
Task Arithmetic [18]	75.94	79.80
Ties Merging [41]	76.18(^{↑ 0.24%})	79.28(^{↓ 0.52%})
Breadcrumbs [4]	76.00(^{↑ 0.06%})	79.60(^{↓ 0.20%})
AdaMerging [43]	77.93(^{↑ 1.99%})	82.83(^{↑ 3.03%})
PCB-Merging [8]	77.19(^{↑ 1.25%})	80.79(^{↑ 0.99%})
KNOTS [33]	76.51(^{↑ 0.57%})	80.95(^{↑ 1.05%})
TSVM [10]	77.19(^{↑ 1.25%})	80.86(^{↑ 1.06%})
CoPA-Merging [47]	76.79(^{↑ 0.85%})	81.43(^{↑ 1.63%})
DO-Merging(ours)	79.24(^{↑ 3.30%})	84.58(^{↑ 4.78%})

Table 4: Multi-task performance when merging T5-large on eight language tasks.

Method	COLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
Pre-trained	73.7	56.6	82.4	91.1	85.5	85.6	94.3	87.5	82.1
Finetune	80.2	88.5	89.2	94.4	87.2	91.7	95.2	90.9	89.6
Task Arithmetic [18]	76.9	85.4	85.3	93.9	85.8	88.1	95.2	87.8	87.3
Ties Merging [41]	77.1	85.1	86.3	93.9	86.0	87.7	95.1	88.0	87.4
BreadCrumbs [4]	77.2	85.2	86.1	93.9	86.0	87.4	94.9	88.4	87.4
PCB-Merging [8]	77.3	85.4	85.8	93.9	86.0	87.9	95.2	88.0	87.4
KNOTS [33]	76.8	87.4	86.4	93.5	86.3	87.3	95.2	88.6	87.7
TSVM [10]	76.0	88.0	84.3	92.9	86.3	87.4	95.2	88.5	87.3
CoPA-Merging [47]	76.0	88.4	86.5	93.0	86.1	87.7	95.3	88.4	87.7
DO-Merging(ours)	78.9	88.5	88.2	93.8	86.5	87.4	96.1	89.2	88.6

Table 5: The merging performance of LLaMa3-8B and Qwen-14B on the corresponding tasks.

Base Model	LLaMa3-8B							Qwen-14B			
	Task	SNLI	MNLI	SICK	QNLI	RTE	SCITAIL	Avg.	MMLU	TruthfulQA	BBQ
Pre-trained	41.69	34.64	55.63	51.92	42.03	60.72	47.77	69.30	51.27	80.69	67.09
Finetune	92.49	90.30	91.58	94.48	89.85	96.51	92.53	68.35	54.34	93.53	72.07
Task Arithmetic [18]	86.56	86.05	80.57	64.91	89.85	93.36	83.55	67.62	53.38	78.24	66.41
Ties Merging [41]	86.48	86.15	80.43	65.04	89.85	93.75	83.61	68.27	50.01	84.10	67.46
KNOTS [33]	87.95	86.94	83.03	67.20	89.58	93.58	84.71	68.20	52.48	83.56	68.08
TSVM [10]	88.82	83.96	81.79	74.87	91.30	94.02	85.79	68.10	52.24	82.42	67.58
CoPA-Merging [47]	88.59	87.49	84.59	67.60	89.85	93.67	85.29	68.10	52.11	83.75	67.98
DO-Merging(ours)	89.05	88.05	83.14	77.13	91.30	94.02	87.11	68.45	53.88	84.99	69.10

These tasks include COLA, MNLI, MRPC, QNLI, QQP, RTE, SST2, and STSB [34]. Detailed fine-tuning information can be found in Appendix C.1. The detailed experimental results are shown in Tab. 3 and Tab. 4. It can be observed that for both T5-base and T5-large, our DO-Merging method shows an improvement of over 1%. This aligns with the analysis of our method’s advantages, indicating that it generalizes well to language models and confirming its strong adaptability.

Additionally, we perform experiments on large-scale language models. We select LLaMa-3-8B [12] and Qwen-14B as base models. For LLaMa-3-8B, we use six natural language understanding tasks as target tasks [33], while for Qwen-14B [1], we chose MMLU [13], TruthfulQA [21], and BBQ [28] as target tasks. Detailed fine-tuning information is provided in Appendix C.1. The experimental results are presented in Tab. 5. In the experiments on LLaMa3-8B, our proposed method shows an improvement of nearly 2% on average, while existing merging methods do not offer significant gains over the basic baseline, Task Arithmetic. Similarly, in the Qwen-14B experiments, DO-Merging achieves more than a 1% average improvement. This aligns with our analysis that addressing the performance loss due to magnitude differences in LoRA merging is crucial. This performance enhancement further confirms the superior performance and generalization ability of DO-Merging.

4.3 Multi-modal Models

To further evaluate the generalization ability of DO-Merging, we conduct experiments on multi-modal tasks. We use Qwen2-VL [36] as the base model and fine-tune then test it on five multi-modal tasks: POPE [20], MMStar [3], MMBench [23], RealWorldQA [39], and MathVista [24]. Detailed weight information can be found in the Appendix. Among these tasks, POPE focuses on detecting model hallucinations, while MMStar and MMBench evaluate different aspects of multi-modal capabilities from diverse metrics. RealWorldQA requires the model to understand real-world scenarios, and MathVista tests its mathematical reasoning ability. These tasks vary significantly in terms of required skills, making them relatively challenging for model merging. As shown in Tab. 6, our DO-Merging method achieves the best performance across all tasks, and even outperforms the fine-tuned model on MMStar. This demonstrates that our method can be effectively applied to multi-modal tasks. Moreover, by selecting tasks with low correlation, we verify that DO-Merging still improves merging performance, further confirming its strong generalization capability and superior performance.

Table 6: Multi-task performance when merging Qwen2-VL on multi-modal tasks.

Method	POPE	MMStar	MathVista	RealworldQA	MMBench	Avg.
Pre-trained	85.86	59.33	57.80	69.80	80.32	70.62
Finetune	97.75	63.60	64.00	73.46	97.50	79.26
Task Arithmetic [18]	91.27	64.60	61.50	69.67	90.98	75.60
Ties Merging [41]	91.27	64.80	61.40	70.42	91.13	75.80(↑ 0.20%)
Breadcrumbs [4]	92.83	66.40	60.69	69.67	90.98	76.11(↑ 0.51%)
KNOTS [33]	92.42	65.64	61.55	70.46	92.58	76.53(↑ 0.93%)
TSVM [10]	92.74	65.26	61.60	70.21	93.49	76.66(↑ 1.06%)
CoPA-Merging [47]	93.32	65.33	62.50	70.31	93.45	76.98(↑ 1.38%)
DO-Merging(ours)	94.74	66.80	62.80	72.13	96.30	78.50(↑ 2.90%)

Table 7: Ablation study on the proposed decoupling and orthogonal components. \times indicates *matrix norm* uses the full parameter matrix's norm that the module is not included, while \checkmark indicates that the module is included.

Table 8: Ablation study on magnitude vectors: normalization coefficients, while *row norm* and *col norm* use row-wise and column-wise coefficients.

Decouple	Orthogonalize	ViT-B/32			ViT-B/16			ViT-L/14			ViT-B/32			ViT-B/16			ViT-L/14		
		Task Arithmetic	Matrix Norm	Row Norm	Col Norm	Task Arithmetic	Matrix Norm	Row Norm	Col Norm	Task Arithmetic	Matrix Norm	Row Norm	Col Norm	Task Arithmetic	Matrix Norm	Row Norm	Col Norm		
\times	\times	74.06	75.94	79.80						74.06	75.94	79.80							
\times	\checkmark	76.90	78.07	83.02						76.98	76.21	77.45							
\checkmark	\times	74.93	77.05	83.20						76.89	76.67	82.52							
\checkmark	\checkmark	77.87	79.24	84.58						77.87	79.24	84.58							

4.4 Ablation Study

In this section, we conduct ablation studies on our method.

Decouple and Orthogonalize. We present ablation results for combining the two components we proposed in DO-Merging. We perform experiments on ViT-B/32, ViT-B/16, and ViT-L/14, using eight visual classification tasks as target tasks. The detailed results are shown in Tab. 7. Compared to methods that include neither component, adding only the orthogonal part leads to an average improvement of over 2%, while adding only the decoupling part results in an improvement of over 1%. This holds true across all three ViT models. When both components are used together, the performance gain is even larger. This strongly demonstrates the effectiveness of both components in DO-Merging, which aligns with our previous theoretical and experimental analysis.

Magnitude Extraction. In Tab. 8, we show results for using different parts of the parameter matrix as the magnitude vector in decoupling. *Matrix Norm* means using the normalization coefficient of the entire parameter matrix as the magnitude vector, while *Row Norm* and *Col Norm* refer to using the row-wise and column-wise normalization coefficients, respectively. From the comparison, we observe that using the matrix norm does not perform well, likely due to its coarse granularity, which fails to preserve fine-grained features during merging. Consistent with our analysis, using the column norm achieves better performance than using the row norm. This is because, in a parameter matrix, each column is responsible for one output dimension, and during merging, we aim to keep the merged output as close as possible to the original models' outputs. Therefore, it is reasonable to use the column norm as the magnitude vector in our decoupling design.

4.5 Discussions

We provide a broader discussion of our DO-Merging. More discussion is in Appendix D.

Flexible Combination. We discuss how our decoupling and orthogonal methods can be flexibly combined with other model merging approaches to further improve their performance. We integrate our proposed components with two popular existing methods, Breadcrumbs and Ties-Merging, and conduct experiments on ViT-B/32. Detailed results are shown in Fig. 4(a). It can be observed that whether adding only the decoupling component or only the orthogonal component, our method consistently improves the performance of existing merging techniques. Both components demonstrate strong compatibility with different methods, allowing users to choose based on practical needs or

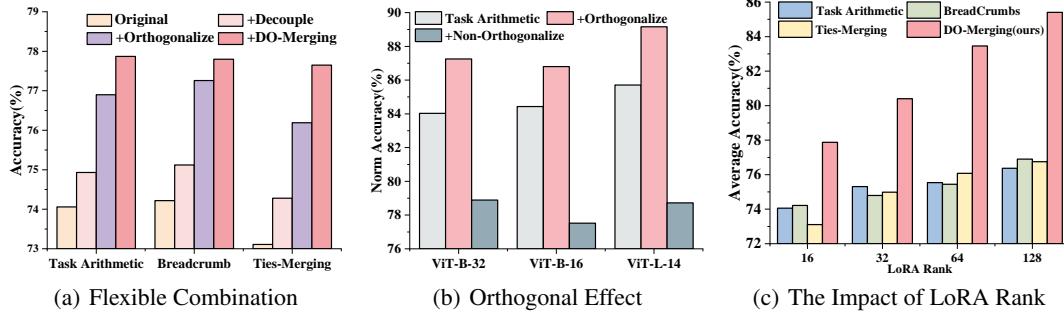


Figure 4: Discussion on Key Properties of DO-Merging. (a). Both components of DO-Merging can be freely combined with other merging methods, and bring near free-lunch improvement. (b). Orthogonality in LoRA is crucial. Performance drops significantly when the direction vectors are made non-orthogonal. (c). Our method performs well across different LoRA ranks.

preferred base model merging strategies. This highlights the flexibility of our decoupling and orthogonal approaches, which significantly benefits the applicability.

Orthogonal Effect. Here, we examine whether our proposed orthogonal method plays a key role in improving merging performance. In Fig. 4(b), we compare the effectiveness of our orthogonal approach by using gradient descent and gradient ascent during the optimization process. We find that when the orthogonality between different LoRA modules is reduced (i.e., increased non-orthogonality), the average accuracy of LoRA merging drops significantly. This strongly indicates that orthogonality is a decisive factor in maintaining performance during model merging. Therefore, our design of enforcing orthogonality among LoRA parameters is well justified.

The Impact of LoRA Rank. Most experiments in the previous sections used a LoRA rank of 16, a common configuration. Here, we evaluate our method’s effectiveness across various ranks. Using ViT-B/32 as the base model, we tested DO-Merging on eight visual tasks with LoRA ranks of 16, 32, 64, and 128. The results in Fig. 4(c) show that our method consistently outperforms popular model merging techniques across all ranks. Additionally, performance improves with increasing rank, demonstrating the robustness and superiority of our approach.

Transfer to Full-Finetune Merging. In this part, we evaluate its effectiveness of DO-Merging when transferred to full-finetune merging. We use ViT-B/32 as the base model and test on eight vision tasks. The results are shown in Fig. 4(c). It can be observed that orthogonalization remains highly effective. And the gain from decoupling is less significant compared to LoRA merging, which aligns with our observation that full fine-tuning leads to smaller magnitude differences across models. Nevertheless, our method still performs well under full fine-tuning, demonstrating its generalization ability.

5 Conclusion

In this paper, we address the issue that existing model merging methods do not perform well when directly applied to LoRA. We find that this is mainly due to great parameter magnitude differences that commonly arise during LoRA training. To tackle this problem, we propose a LoRA merging method in which we decouple the magnitude and direction of the parameters. This helps reduce the loss of directional information during merging caused by magnitude variations. Furthermore, to reduce task conflicts, we apply data-free orthogonal perturbations to the direction vectors. Based on these ideas, we propose our DO-Merging. This approach does not require access to training data and can be flexibly used in different combinations, making it a promising solution for LoRA merging.

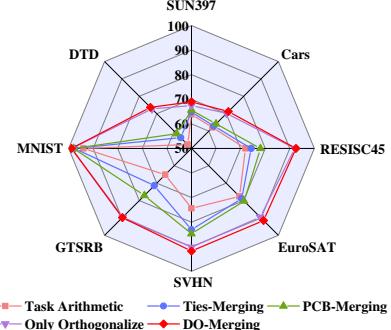


Figure 5: Applying DO-Merging to the merging of fully fine-tuned models is also effective. Experiments are conducted on ViT-B/32.

References

- [1] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.
- [2] Hongxu Chen, Runshi Li, Bowei Zhu, Zhen Wang, and Long Chen. Iteris: Iterative inference-solving alignment for lora merging. *arXiv preprint arXiv:2411.15231*, 2024.
- [3] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.
- [4] MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks. *arXiv preprint arXiv:2312.06795*, 2023.
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.
- [7] Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, et al. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 4(7), 2023.
- [8] Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim Kuan Goh, Ho-Kin Tang, Daojing He, et al. Parameter competition balancing for model merging. *arXiv preprint arXiv:2410.02396*, 2024.
- [9] Shangbin Feng, Zifeng Wang, Yike Wang, Sayna Ebrahimi, Hamid Palangi, Lesly Miculicich, Achin Kulshrestha, Nathalie Rauschmayr, Yejin Choi, Yulia Tsvetkov, et al. Model swarms: Collaborative search to adapt llm experts via swarm intelligence. *arXiv preprint arXiv:2410.11163*, 2024.
- [10] Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodolà. Task singular vectors: Reducing task interference in model merging. *arXiv preprint arXiv:2412.00081*, 2024.
- [11] Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodolà. Task singular vectors: Reducing task interference in model merging. *arXiv preprint arXiv:2412.00081*, 2024.
- [12] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- [13] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.
- [14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- [15] Xiaolin Hu, Xiang Cheng, Peiyu Liu, Wei Liu, Jian Luan, Bin Wang, and Yong Liu. Dota: Weight-decomposed tensor adaptation for large language models. *arXiv preprint arXiv:2412.20891*, 2024.

- [16] Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition. In *First Conference on Language Modeling*.
- [17] Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. Emr-merging: Tuning-free high-performance model merging. *arXiv preprint arXiv:2405.17461*, 2024.
- [18] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhad. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.
- [19] Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.
- [20] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.
- [21] Stephanie Lin, Jacob Hilton, and Owain Evans. 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, 2022.
- [22] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.
- [23] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer, 2024.
- [24] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *The Twelfth International Conference on Learning Representations*.
- [25] Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. Twin-merging: Dynamic integration of modular expertise in model merging. *arXiv preprint arXiv:2406.15479*, 2024.
- [26] Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.
- [27] Ziheng Ouyang, Zhen Li, and Qibin Hou. K-lora: Unlocking training-free fusion of any subject and style loras. *arXiv preprint arXiv:2502.18461*, 2025.
- [28] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel Bowman. Bbq: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2086–2105, 2022.
- [29] Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. Lora soups: Merging loras for practical skill composition tasks. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 644–655, 2025.
- [30] Biqing Qi, Fangyuan Li, Zhen Wang, Junqi Gao, Dong Li, Peng Ye, and Bowen Zhou. Less is more: Efficient model merging with binary task switch. *arXiv preprint arXiv:2412.00054*, 2024.
- [31] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

- [32] Donald Shenaj, Ondrej Bohdal, Mete Ozay, Pietro Zanuttigh, and Umberto Michieli. Lora-rar: Learning to merge loras via hypernetworks for subject-style conditioned image generation. *arXiv preprint arXiv:2412.05148*, 2024.
- [33] George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with svd to tie the knots. In *The Thirteenth International Conference on Learning Representations*.
- [34] Anke Tang, Li Shen, Yong Luo, Han Hu, Bo Du, and Dacheng Tao. Fusionbench: A comprehensive benchmark of deep model fusion. *arXiv preprint arXiv:2406.03280*, 2024.
- [35] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [36] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.
- [37] T Wolf. Huggingface’s transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.
- [38] Xun Wu, Shaohan Huang, and Furu Wei. Mixture of lora experts. In *The Twelfth International Conference on Learning Representations*.
- [39] xAI. Grok 3: The most powerful AI in the world is here, February 2025.
- [40] Feng Xiong, Runxi Cheng, Wang Chen, Zhanqiu Zhang, Yiwen Guo, Chun Yuan, and Ruifeng Xu. Multi-task model merging via adaptive weight disentanglement. *arXiv preprint arXiv:2411.18729*, 2024.
- [41] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36, 2024.
- [42] Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*, 2024.
- [43] Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*, 2024.
- [44] Yang Yang, Wen Wang, Liang Peng, Chaotian Song, Yao Chen, Hengjia Li, Xiaolong Yang, Qinglin Lu, Deng Cai, Boxi Wu, et al. Lora-composer: Leveraging low-rank adaptation for multi-concept customization in training-free diffusion models. *arXiv preprint arXiv:2403.11627*, 2024.
- [45] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Extend model merging from fine-tuned to pre-trained large language models via weight disentanglement. *arXiv preprint arXiv:2408.03092*, 2024.
- [46] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024.
- [47] Fanhu Zeng, Haiyang Guo, Fei Zhu, Li Shen, and Hao Tang. Parameter efficient merging for multimodal large language models with complementary parameter adaptation. *arXiv preprint arXiv:2502.17159*, 2025.
- [48] Juzheng Zhang, Jiacheng You, Ashwinee Panda, and Tom Goldstein. Lori: Reducing cross-task interference in multi-task low-rank adaptation. *arXiv preprint arXiv:2504.07448*, 2025.

- [49] Yiqun Zhang, Peng Ye, Xiaocui Yang, Shi Feng, Shufei Zhang, Lei Bai, Wanli Ouyang, and Shuyue Hu. Nature-inspired population-based evolution of large language models. *arXiv preprint arXiv:2503.01155*, 2025.
- [50] Yuxuan Zhang and Ruizhe Li. Dlp-lora: Efficient task-specific lora fusion with a dynamic, lightweight plugin for large language models. *arXiv preprint arXiv:2410.01497*, 2024.
- [51] Xinyu Zhao, Guoheng Sun, Ruisi Cai, Yukun Zhou, Pingzhi Li, Peihao Wang, Bowen Tan, Yexiao He, Li Chen, Yi Liang, et al. Model-glue: Democratized llm scaling for a large model zoo in the wild. *Advances in Neural Information Processing Systems*, 37:13349–13371, 2024.
- [52] Ziyu Zhao, Tao Shen, Didi Zhu, Zexi Li, Jing Su, Xuwu Wang, Kun Kuang, and Fei Wu. Merging loras like playing LEGO: pushing the modularity of lora to extremes through rank-wise clustering. *CoRR*, abs/2409.16167, 2024.
- [53] Shenghe Zheng and Hongzhi Wang. Free-merging: Fourier transform for model merging with lightweight experts. *arXiv preprint arXiv:2411.16815*, 2024.
- [54] Yu Zhou, Xingyu Wu, Jibin Wu, Liang Feng, and Kay Chen Tan. Hm3: Hierarchical multi-objective model merging for pretrained models. *arXiv preprint arXiv:2409.18893*, 2024.

Appendix for DO-Merging

A Notations

Table 9: Notations.

Notations	Descriptions
θ_{pre}	Pre-trained model parameters.
Δ_i	Fine-tuning parameters for task i .
W_{pre}^k	The k -th layer of fine-tuning parameters for task i .
W_i^k	The k -th layer of fine-tuning parameters for task i .
A_i, B_i	The k -th layer of LoRA low-rank parameters for task i .
\hat{W}_i^k	The k -th layer of fine-tuning parameters for task i after orthogonalization.
\bar{W}_i^k	The k -th layer of fine-tuning direction parameters for task i after decoupling.
α_i^k	The k -th layer of fine-tuning magnitude parameters for task i after decoupling.
W_{ij}	The row- i and col- j of a parameter matrix.

Tab. 9 presents the main notations used in this paper and their corresponding meanings. It should be noted that, for simplicity in explanation and proof, the layer index k is sometimes omitted when it does not affect the clarity of the discussion or derivation.

B Proofs

In this section, we provide a proof for the theorem presented in the original paper.

B.1 Theorem 3.1

It follows from the Assumption 3.1 that:

$$W_1 = \alpha_1 \bar{W}_1, W_2 = \alpha_2 \bar{W}_2 \quad (6)$$

The simplest merging model can be expressed as:

$$W = \frac{1}{2}(W_1 + W_2) \quad (7)$$

Let $\|\alpha_1\|_2 = \lambda \|\alpha_2\|_2$, Then we have:

$$L = \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_1\|_2} \|W - W_1\|_2 + \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_2\|_2} \|W - W_2\|_2 \quad (8)$$

$$= \frac{\lambda^2 + 1}{\lambda^2} \|\alpha_2\|_2 \left\| \frac{1}{2}(W_2 - W_1) \right\|_2 + (\lambda^2 + 1) \|\alpha_2\|_2 \left\| \frac{1}{2}(W_2 - W_1) \right\|_2 \quad (9)$$

Because $\bar{W}_{1ij} \sim N(0, 1)$, and $\bar{W}_{2ij} \sim N(0, 1)$.

Then we have:

$$\mathbb{E}(L) = \mathbb{E}\{(\lambda^2 + 1) \left[\sum_{i=1}^m \sum_j \frac{1}{\lambda^2} \left(\frac{1}{4} \alpha_{1j}^2 (\bar{W}_1)_{ij}^2 + \frac{1}{4} \alpha_{2j}^2 (\bar{W}_2)_{ij}^2 - \frac{1}{2} \alpha_{1j} \alpha_{2j} (\bar{W}_1)_{ij} (\bar{W}_2)_{ij} \right) \right] \} \quad (10)$$

$$+ \left(\frac{1}{4} \alpha_{1j}^2 (\bar{W}_1)_{ij}^2 + \frac{1}{4} \alpha_{2j}^2 (\bar{W}_2)_{ij}^2 - \frac{1}{2} \alpha_{1j} \alpha_{2j} (\bar{W}_1)_{ij} (\bar{W}_2)_{ij} \right) \} \quad (11)$$

$$= (\lambda^2 + 1) \left[\sum_{i=1}^m \sum_j \frac{1}{4} \left(1 + \frac{1}{\lambda^2} \right) (\alpha_{1j}^2 + \alpha_{2j}^2) \right] \quad (12)$$

$$= n(\lambda^2 + 1) \|\alpha_2\|_2 \left[\left(\frac{1}{\lambda^2} + 1 \right) \left(\frac{1}{4} \lambda^2 + \frac{1}{4} \right) \right] \quad (13)$$

$$= n(\lambda^2 + 1) \|\alpha_2\|_2 \left(\frac{1}{4} \lambda^2 + \frac{1}{2} + \frac{1}{4\lambda^2} \right) \quad (14)$$

Taking the derivative of Equation 10 with respect to λ , we obtain:

$$f'(\lambda) = n(\lambda^2 + 1)\|\alpha_2\|_2\left(\frac{1}{2}\lambda - \frac{1}{2\lambda^3}\right) \quad (15)$$

When $\lambda = 1$, we have $f'(\lambda) = 0$. When $\lambda < 1$, $f'(\lambda) < 0$; and when $\lambda > 1$, $f'(\lambda) > 0$. Therefore, f reaches its minimum at $\lambda = 1$. In other words, the expression is minimized when $\|\alpha_1\|_2 = \|\alpha_2\|_2$, which completes the proof.

Note that this analysis assumes the merging of two models. However, the conclusion can be naturally extended to the case of merging multiple models.

B.2 Theorem 3.2

For methods without decoupling, we have: $W = \frac{1}{2}(W_1 + W_2)$.

For methods with decoupling, we have: $W = \frac{1}{4}(\alpha_1 + \alpha_2)(\overline{W_1} + \overline{W_2})$.

Assume $\|\alpha_2\|_2 = \lambda^2\|\alpha_1\|_1, \lambda > 1$. Also note that $\forall \alpha \in \alpha_1, \alpha \geq 0$.

For Case 1, we have:

$$L = \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_1\|_2}\|W - W_1\|_2 + \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_2\|_2}\|W - W_2\|_2 \quad (16)$$

$$= \frac{\lambda^2 + 1}{\lambda^2}\|\alpha_2\|_2\left(\frac{1}{2}(W_2 - W_1)\right)\|_2 + (\lambda^2 + 1)\|\alpha_2\|_2\left(\frac{1}{2}(W_2 - W_1)\right)\|_2 \quad (17)$$

Then we have:

$$\mathbb{E}(L) = \mathbb{E}\{(\lambda^2 + 1)[\sum_{i=1}^m \sum_j \frac{1}{\lambda^2}(\frac{1}{4}\alpha_{1j}^2(\overline{W_1})_{ij}^2 + \frac{1}{4}\alpha_{2j}^2(\overline{W_2})_{ij}^2 - \frac{1}{2}\alpha_{1j}\alpha_{2j}(\overline{W_1})_{ij}(\overline{W_2})_{ij})] \quad (18)$$

$$+ (\frac{1}{4}\alpha_{1j}^2(\overline{W_1})_{ij}^2 + \frac{1}{4}\alpha_{2j}^2(\overline{W_2})_{ij}^2 - \frac{1}{2}\alpha_{1j}\alpha_{2j}(\overline{W_1})_{ij}(\overline{W_2})_{ij})\} \quad (19)$$

$$= (\lambda^2 + 1)[\sum_{i=1}^m \sum_j \frac{1}{4}(1 + \frac{1}{\lambda^2})(\alpha_{1j}^2 + \alpha_{2j}^2)] \quad (20)$$

$$= n(\lambda^2 + 1)\|\alpha_2\|_2[(\frac{1}{\lambda^2} + 1)(\frac{1}{4}\lambda^2 + \frac{1}{4})] \quad (21)$$

$$= n(\lambda^2 + 1)\|\alpha_2\|_2(\frac{1}{4}\lambda^2 + \frac{1}{2} + \frac{1}{4\lambda^2}) \quad (22)$$

For Case 2, we have:

$$L = \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_1\|_2}\|W - W_1\|_2 + \frac{\|\alpha_1\|_2 + \|\alpha_2\|_2}{\|\alpha_2\|_2}\|W - W_2\|_2 \quad (23)$$

$$= \frac{\lambda^2 + 1}{\lambda^2}\|\alpha_2\|_2\left(\frac{1}{4}(\alpha_1 + \alpha_2)(\overline{W_1} + \overline{W_2}) - W_1\right)\|_2 \quad (24)$$

$$+ (\lambda^2 + 1)\|\alpha_2\|_2\left(\frac{1}{4}(\alpha_1 + \alpha_2)(\overline{W_1} + \overline{W_2}) - W_2\right)\|_2 \quad (25)$$

Then we have:

$$\mathbb{E}(L_2) = \mathbb{E}\{(\lambda^2 + 1)[\sum_{i=1}^m \sum_j \frac{1}{\lambda^2}(W_{ij}^2 - (W_1)_{ij}^2) + (W_{ij}^2 - (W_2)_{ij}^2)]\} \quad (26)$$

$$= (\lambda^2 + 1)[\sum_{i=1}^m \sum_j \frac{1}{\lambda^2}(\frac{1}{8}(\alpha_1 + \alpha_2)^2 + \frac{1}{2}\alpha_1^2 - \frac{1}{2}\alpha_1\alpha_2)] \quad (27)$$

$$+ (\frac{1}{8}(\alpha_1 + \alpha_2)^2 + \frac{1}{2}\alpha_2^2 - \frac{1}{2}\alpha_1\alpha_2)] \quad (28)$$

$$= n(\lambda^2 + 1)\|\alpha_2\|_2[\frac{1}{\lambda^2}(\frac{5}{8}\lambda^2 - \frac{1}{4}\lambda + \frac{1}{8}) + \frac{1}{8}\lambda^2 - \frac{1}{4}\lambda + \frac{5}{8}] \quad (29)$$

$$= n(\lambda^2 + 1)\|\alpha_2\|_2(\frac{1}{8}\lambda^2 - \frac{1}{4}\lambda + \frac{5}{4} - \frac{1}{4\lambda} + \frac{1}{8\lambda^2}) \quad (30)$$

Then we have:

$$f(\lambda) = \mathbb{E}(L_1) - \mathbb{E}(L_2) \quad (31)$$

$$= n(\lambda^2 + 1)\|\alpha_2\|_2\left(\frac{1}{8}\lambda^2 + \frac{1}{4}\lambda - \frac{3}{4} + \frac{1}{4\lambda} + \frac{1}{8\lambda^2}\right). \quad (32)$$

We have $f(1) = 0$, and:

$$f'(\lambda) = n(\lambda^2 + 1)\|\alpha_2\|_2\left(\frac{1}{4}\lambda - \frac{1}{4\lambda^2} - \frac{1}{4\lambda^3} + \frac{1}{4}\right) \quad (33)$$

When $\lambda > 1$, we have $f'(\lambda) > 0$, and $f(1) = 0$. Therefore, for $\lambda > 1$, it holds that $f(\lambda) > 0$, which implies Equation 1 > Equation 2. This completes the proof.

B.3 Theorem 3.3

We begin by defining **parameter conflict**. For each position, parameter conflict mainly refers to the situation where the corresponding elements in the two matrices being merged have opposite signs. Specifically, for merged matrices W_1 and W_2 , we say there is a parameter conflict at position (i, j) if:

$$\text{sign}((W_1)_{ij}) \neq \text{sign}((W_2)_{ij}). \quad (34)$$

We now proceed to prove Theorem 3.3. Without loss of generality, we simplify the problem to the merging of two matrices, W_1 and W_2 . It suffices to show that as $\|\delta_i\| \rightarrow 0$, a smaller $\|W_1^T W_2\|$ leads to fewer parameter conflicts during merging.

To this end, we focus on the parameter conflict at a specific position (i, j) , i.e., the relationship between W_1_{ij} and W_2_{ij} . Assume $\|W_1_{ij}\| > \|W_2_{ij}\|$. To simplify the proof, we consider applying only an orthogonal perturbation to W_1_{ij} . The orthogonal perturbation is governed by the following loss function:

$$L_o = \|W_1^T (W_2 + \delta)\| \quad (35)$$

where δ denotes the orthogonal perturbation applied to W_2 .

With the constraint $\|\delta\| \rightarrow 0$. Then we have:

$$\frac{\partial L_o}{\partial \delta} = \frac{\|W_1^T (W_2 + \delta)\|}{\|\delta\|} \quad (36)$$

Since we only consider the optimization direction constraint of $(W_1)_{ij}$ on $(W_2)_{ij}$, and $\text{sign}((W_1)_{ij}) \neq \text{sign}((W_2)_{ij})$, the corresponding partial derivative is:

$$\left(\frac{\partial L_o}{\partial \delta}\right)_{ij} = -(W_1)_{ij} \quad (37)$$

The gradient direction is aligned with W_2_{ij} . In other words, during gradient descent on L_o , $(W_2)_{ij}$ tends to move closer to $(W_1)_{ij}$, thereby reducing parameter conflict. This completes the proof.

C Reproducibility

C.1 Datasets

Merging 8 ViTs. We use ViT-B/32 and ViT-L/14 as pre-trained models, and fine-tune them on 8 image classification datasets (SUN397, Cars, RESISC45, EuroSAT, SHVN, GTSRB, MNIST, and DTD), then merge the models and test their performance. Configuration details follow [17].

Merging Medium-sized Language Models. We use T5 as the pre-trained model, fine-tune it on 8 classification task datasets from GLUE benchmark for model merging, including CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI, and RTE. CoLA is evaluated with the Matthews correlation coefficient, STS-B with the average of the Pearson and Spearman correlation coefficients, and the others by accuracy. Details follow [17].

Merging Large Language Models. In this section, we conduct two experiments. The first uses LLaMa3-8B as the pre-trained model and LoRA as the PEFT method, fine-tuned and merged on SNLI, MNLI, SICK, QNLI, RTE, SCITAIL. Details can be found in [33]. The second part uses

Qwen-14B as the pre-trained model and LoRA as the PEFT method. We fine-tune and merging on three generative tasks: MMLU, TruthfulQA, and BBQ. Configuration details can be found in [25].

Merging Multi-Modal Models. In this part, we select Qwen2-VL as the base model and evaluate it on five multimodal tasks: POPE, MMStar, MMBench, MathVista, and RealWorldQA. We also use open-source weights for model merging. The open-source weights are available at provided link¹.

C.2 Baselines

Finetune. Solve each task with its fine-tuned model, but this requires storing separate models for each task, leading to significant storage overhead.

Task Arithmetic [18]. Define task vectors as the merging target. For task k , the task vector is defined as $v_k = \theta_k - \theta_{pre}$, where θ_{pre} is the pre-trained model parameters, and θ_k is the fine-tuned parameters for task k . The merging process can be represented as $\theta_m = \theta_{pre} + \lambda \sum_{i=1}^K v_i$, where λ is the merging coefficient. This method suffers significant performance degradation due to unaddressed task conflicts. For LoRA, the task vector refers to the matrix constructed by LoRA.

Ties-Merging [41]. Attempts to resolve parameter conflicts during model merging by eliminating redundancy and sign conflicts. However, resolving parameter conflicts is insufficient to address task conflicts, resulting in performance loss.

Breadcrumbs [4]. Discards parameters with the largest and smallest absolute values as redundant, negatively impacting model merging. This approach is simple, but it shows a significant performance decline on certain tasks.

PCB-Merging [8]. Uses internal balancing to measure parameter importance within tasks and mutual balancing to assess parameter similarity across tasks, discarding redundant parameters and adjusting merging coefficients. This approach requires considerable computational resources.

AdaMerging [43]. Uses an unsupervised approach to learn the merging coefficient for each task vector or layer. AdaMerging++ additionally applies Ties-Merging before calculating the merging coefficient. This method is limited to classification tasks and requires certain training resources, making it unsuitable for edge deployment.

DARE [46]. DARE randomly discards a large portion of task vector parameters before merging, potentially reducing parameter interference among models. This method is simple, but due to the lack of further optimization in the merging, it suffers from significant performance degradation.

TSVM [10]. The motivation of this method lies in the effectiveness of applying whitening constraints on the SVD components of the task vector before merging. In essence, this serves as a form of orthogonal constraint. However, unlike our approach, this constraint does not guarantee the preservation of the original model’s capabilities under all circumstances. It may affect the model’s initial performance and lead to a decline in merging quality. In contrast, the orthogonal method we propose offers stronger generalization ability.

KNOTS [33]. This is a method specifically designed for LoRA merging. The key observation is that, compared to full fine-tuning, LoRA produces parameter features with greater discrepancies. To address this, the paper proposes merging LoRA models in a shared SVD space, which helps bring their feature outputs closer together. However, this approach does not account for the fact that the issue of uneven distribution in the original space still exists in the shared space. As a result, it fails to address the performance loss caused by the parameter magnitude distribution issues we identify, leading to only limited improvements in merging performance.

COPA-Merging [47]. This is another method designed specifically for LoRA merging. Its main idea is based on the observation that the parameter distributions of LoRA’s A and B matrices exhibit different characteristics. Accordingly, it automatically computes normalization coefficients during merging based on these differences and incorporates them into the computation of merged parameters. While this approach identifies the distributional differences between the A and B matrices in LoRA, it does not further address the more fundamental issue that the distribution gap between LoRA and full fine-tuning which leads to suboptimal performance in LoRA merging. In contrast, our work tackles the root cause of the poor performance in LoRA merging.

¹POPE; MMStar; MMBench ; MathVista ;RealWorldQA.

C.3 Experiment Details

In this section, we discuss the details of the experiments. All experiments are conducted on a single NVIDIA A800 GPU.

C.4 Computational Details

Magnitude Distribution Variance. Assume there are n fine-tuned models $\{\theta_i\}_{i=1}^n$, each containing K layers. The k -th layer of the i -th model is denoted as W_i^k . The Magnitude Distribution Variance v is then defined as:

$$v = \sum_{j=1}^k \text{var}(|W_i^k|_{i=1}^n), \quad (38)$$

where $\text{var}(\cdot)$ denotes the variance operator.

Norm Average Accuracy. Assume we have n fine-tuned models corresponding to n tasks $\{t_i\}_{i=1}^n$, and the performance of these fine-tuned models on their respective tasks is $\{a_i\}_{i=1}^n$. Let the performance of the merged model on these n tasks be $\{\hat{a}_i\}_{i=1}^n$. Then, the normalized average performance (acc_n) of the merged model is defined as:

$$acc_n = \frac{\sum_{i=1}^n \hat{a}_i}{\sum_{i=1}^n a_i}. \quad (39)$$

D More Discussions

In this part, we provide a more detailed explanation of our method. In Sec. D.1, we introduce the key innovations of our approach in greater detail. In Sec. D.2 and Sec. D.3, we analyze why we choose model merging instead of other techniques to enhance LoRA’s multi-task capabilities. In Sec. D.3, we explain why we merge the product matrix of LoRA’s two low-rank matrices instead of merging the matrices directly.

D.1 Novelty of DO-Merging

In this section, we analyze the key innovations of our proposed DO-Merging method, which are summarized in three aspects.

Observation. As shown in Fig. 1, we observe that LoRA modules trained on different tasks exhibit significantly larger variations in parameter magnitude compared to full fine-tuning. We further establish a connection between this observation and the suboptimal performance of existing merging methods on LoRA, both empirically and theoretically. **To the best of our knowledge, this is the first work that analyzes the parameter difference between LoRA and full fine-tuning from this perspective and proposes targeted improvements for LoRA merging.**

Decouple. Based on the phenomenon, we demonstrate the correlation between parameter distribution and merging performance. **We argue that effective model merging should preserve both the shape and scale of parameter distributions.** When the scale differences are large, parameters with larger magnitudes tend to dominate the merged result, leading to information loss from smaller-scale parameters and thus performance degradation. To address this, we propose a decoupling strategy. Unlike previous decoupling methods that are typically applied during training or used to compute merging coefficients, our approach does not impose constraints on the training process. This allows us to make full use of publicly available LoRA weights and improves usability. Compared to coefficient-based approaches, we explore how weight decoupling can be effectively applied during the merging process, supported by theoretical analysis and empirical results.

Orthogonalize. Our orthogonalization strategy aims to reduce task interference during the merging of direction vectors. Prior works have shown that enforcing orthogonality among different LoRA modules during training can improve merging performance [48]. However, since most publicly available LoRA weights do not satisfy such constraints, we propose an orthogonalization method applicable to pre-trained weights. To meet the efficiency needs of diverse users, we introduce a **data-free and layer-wise** orthogonalization approach. Unlike prior methods, **we further incorporate constraints on parameter variation ranges to ensure that the orthogonalization process does**

not harm the model’s original task performance or erase task-specific features. We also provide rigorous theoretical guarantees and comprehensive experimental validation for the proposed method.

In summary, we begin by identifying the parameter distribution discrepancy between LoRA and full fine-tuning, and establish its link to the poor performance of LoRA merging through both theory and experiments. To address this issue, we innovatively propose a method that decouples the direction and magnitude of parameter updates. Furthermore, to reduce task interference, we introduce a data-free orthogonalization scheme. For each component of our method, we provide solid theoretical analysis and extensive experimental verification. Therefore, we believe that our DO-Merging method offers a promising and innovative solution for model merging in the LoRA domain.

D.2 Why not LoRA-MoE?

LoRA-MoE [38, 7, 50, 51] is a method that combines multiple LoRA weights using a router, which directs incoming inputs to the corresponding LoRA based on the task. The key advantage of this approach is that, as long as the router performs well, each task-specific LoRA can retain its performance on the corresponding task. However, compared to the model merging approach studied in this paper, LoRA-MoE suffers from two major issues that hinder its practical deployment:

Usability. LoRA-MoE inevitably requires modifications to the model architecture. At a minimum, adding a router and supporting multiple LoRA modules. This poses a significant barrier for users without deep learning expertise who still wish to apply deep learning techniques to their problems. Even for those who are capable of modifying the model, distributing or further fine-tuning the model becomes much more complex. In contrast, the model merging methods we focus on do not alter the model structure and result in a single, standard model. This makes it significantly easier for users to deploy, distribute, and continue training. In today’s context, usability plays a critical role in determining the practical value of a method.

Cost. LoRA-MoE requires training the router, which involves a substantial amount of data and computational resources for both forward and backward passes. When dealing with large models such as 32B or 70B variants, this training becomes practically infeasible for average users due to resource constraints, limiting the widespread adoption of LoRA-MoE.

In summary, LoRA-MoE is better suited for high-performance-demanding scenarios, but its usability and training cost issues limit its broader applicability. On the other hand, model merging is more user-friendly, making it more accessible to a wider audience. Both approaches are valuable, and there is also potential for them to be used together to further enhance performance.

D.3 Why not Model Swarm?

Model Swarm [9, 49] refers to a family of methods that use evolutionary algorithms or other techniques to collaboratively combine existing model weights into stronger models. The advantage of this approach lies in its ability to achieve strong performance, sometimes even outperforming individually fine-tuned models. However, compared to the model merging approach studied in this paper, Model Swarm faces two major issues that limit its practical deployment:

Cost. Model Swarm requires a certain amount of data and substantial computation to determine the direction of collaborative evolution. At the very least, it involves repeated evaluations of each evolved model’s performance to guide further optimization. This process demands a significant amount of inference resources, making it impractical for average users due to high costs. In contrast, the model merging methods we focus on do not require extensive inference to construct a unified weight set, resulting in lower overhead and broader applicability.

Usability. Constructing stronger weights using Model Swarm also requires a certain level of deep learning expertise. As more and more fields adopt deep learning techniques, this requirement becomes a barrier for a large number of non-expert users, limiting the method’s real-world usage.

In summary, Model Swarm suffers from high computational cost and limited usability. It is better suited for performance-critical scenarios, whereas model merging is more appropriate for resource-constrained settings commonly faced by general users. Both approaches are valuable, and there is also potential for them to be combined for further performance gains.

Table 10: Merging results on eight vision tasks using ViT-B/32. *Separate* denotes merging the low-rank matrices individually, while *Concat* denotes merging the product of the low-rank matrices.

Method		SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Task Arithmetic	Separate	61.37	51.08	63.63	70.33	68.73	51.77	85.73	46.60	62.40
	Concat	63.38	60.27	75.46	87.70	84.73	70.74	96.64	53.55	74.06
Ties-Merging	Separate	61.37	51.23	63.56	69.43	68.41	51.97	85.68	46.10	62.21
	Concat	63.81	53.95	73.56	86.00	88.27	71.15	97.81	50.32	73.10
Breadcrumbs	Separate	62.46	52.69	62.51	63.56	64.80	47.36	84.49	45.85	60.46
	Concat	64.02	57.87	74.12	88.73	84.24	74.28	97.29	53.23	74.22

Table 11: Merging results on eight vision tasks using ViT-B/16. *Separate* denotes merging the low-rank matrices individually, while *Concat* denotes merging the product of the low-rank matrices.

Method		SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Task Arithmetic	Separate	64.18	57.32	69.03	63.63	76.73	58.84	94.54	46.38	66.33
	Concat	66.82	63.34	76.32	83.96	91.59	76.90	97.76	50.85	75.94
Ties-Merging	Separate	64.34	57.12	69.64	63.57	76.96	58.46	94.42	46.40	66.36
	Concat	66.41	63.41	77.07	84.06	91.97	77.50	98.02	51.06	76.19
Breadcrumbs	Separate	65.25	59.59	70.16	68.81	73.68	55.26	93.18	46.91	66.61
	Concat	65.38	64.12	77.49	83.75	91.32	77.20	97.87	50.92	76.00

D.4 Why not Merge A and B Separately?

Here, we discuss why our merging method is performed on the product matrix $W = BA$, rather than separately on the individual low-rank matrices A and B . Suppose we have two LoRA modules, and we select a certain layer for merging. Let their parameters be denoted as A_1, B_1 and A_2, B_2 , respectively. Our current merging strategy is formulated as:

$$W_{\text{merge}} = \lambda(B_1A_1 + B_2A_2). \quad (40)$$

In contrast, if we perform merging separately on the A and B matrices and then construct the full matrix, it can be expressed as:

$$W_{\text{merge}} = \beta(B_1 + B_2)(A_1 + A_2) = \beta(B_1A_1 + B_2A_2) + \underbrace{\beta(B_1A_2 + B_2A_1)}_{\text{Cross term}}. \quad (41)$$

It can be observed that this formulation introduces cross terms between the low-rank matrices of different LoRA modules. Intuitively, due to the distinct features learned during training, these cross terms act as noise and degrade the merging performance.

We validate this observation through experiments on ViT-B/32, ViT-B/16, and ViT-L/14 across eight vision tasks. The detailed results are shown in Tab. 10, Tab. 11, and Tab. 12. The results show that, for the same merging method applied to different models, merging the A and B matrices separately typically leads to an average performance drop of over 10% compared to first computing their matrix product and then merging. This performance degradation validates the significant negative impact of the cross terms on merging effectiveness. Therefore, in practice, we perform merging in the $W = BA$ space, which leads to better performance.

E Limitations and Future Works

This paper introduces DO-Merging, a model merging method specifically designed for LoRA, aiming to address the performance degradation caused by the differences between LoRA and full fine-tuning during the adaptation process. Like many existing model merging approaches, its main limitation lies in the need for users to manually find and select the desired model weights, which can be time-consuming and require significant effort in model retrieval.

Future works can focus on developing a fully user-friendly, end-to-end model merging pipeline. Such a system could automatically handle model discovery, selection, merging method adaptation, and result generation based on user requirements. This would greatly enhance the practical usability of model merging techniques.

Table 12: Merging results on eight vision tasks using ViT-L/14. *Separate* denotes merging the low-rank matrices individually, while *Concat* denotes merging the product of the low-rank matrices.

Method		SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Task Arithmetic	Separate	69.11	78.31	79.06	74.93	85.17	66.41	96.48	56.38	75.73
	Concat	70.19	80.09	82.62	79.41	90.18	77.93	98.12	59.89	79.80
Ties-Merging	Separate	69.54	78.24	79.12	74.24	85.56	66.34	96.65	56.14	75.73
	Concat	70.32	79.26	81.95	76.93	91.58	76.70	98.20	59.36	79.29
Breadcrumbs	Separate	67.62	71.15	77.14	66.37	88.86	67.89	96.71	54.73	73.81
	Concat	70.34	80.84	82.18	77.95	90.59	77.49	98.20	59.22	79.60

Table 13: Multi-task performance when merging ViT-B/16 on eight vision tasks.

ViT-B/16		SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Pretrained	63.80	64.66	66.36	54.59	52.01	43.49	51.70	45.00	55.20	
	75.10	99.14	98.64	99.58	95.42	78.62	75.40	97.69	89.94	
Task Arithmetic	66.82	63.34	76.32	83.96	91.59	76.90	97.76	50.85	75.94	
	66.41	63.41	77.07	84.06	91.97	77.50	98.02	51.06	76.19	
Ties Merging	65.38	64.12	77.49	83.75	91.32	77.20	97.87	50.92	76.00	
	64.67	61.63	78.76	92.74	86.41	91.38	97.21	50.69	77.94	
Breadcrumbbs	64.12	61.54	79.21	92.89	87.12	80.60	97.82	54.23	77.19	
	65.30	59.20	78.29	90.38	88.29	80.51	97.84	52.30	76.51	
AdaMerging	65.60	58.89	78.05	90.81	91.86	82.77	98.10	51.51	77.19	
	65.60	59.20	78.25	90.54	89.58	80.49	98.20	52.49	76.79	
PCB-Merging	65.70	66.61	80.78	93.07	89.27	82.90	97.81	57.79	79.24	

Table 14: Multi-task performance when merging ViT-L/14 on eight vision tasks.

ViT-L/14		SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg
Pre-trained	66.80	77.90	71.37	62.15	58.42	50.54	76.35	55.58	64.89	
	79.64	91.32	96.63	99.03	97.76	99.30	99.72	81.49	93.11	
Task Arithmetic	70.19	80.09	82.62	79.41	90.18	77.93	98.12	59.89	79.80	
	70.32	79.26	81.95	76.93	91.58	76.70	98.20	59.36	79.29	
Ties Merging	70.34	80.84	82.18	77.95	90.59	77.49	98.20	59.22	79.60	
	72.42	83.34	83.56	83.53	93.57	84.53	98.20	63.56	82.83	
Breadcrumbbs	70.13	81.40	81.56	81.03	92.35	81.53	98.26	60.12	80.79	
	70.49	80.30	82.21	80.56	92.65	80.92	98.26	62.27	80.95	
AdaMerging	70.77	79.34	83.35	81.63	92.51	79.94	98.36	61.01	80.86	
	70.54	80.68	83.49	83.62	92.48	80.28	97.84	62.56	81.43	
DO-Merging(ours)	70.22	85.35	87.90	84.92	94.74	87.13	97.78	68.62	84.58	

F Additional Experimental Results

In this section, we present additional experimental results. Tab. 13 shows the test results of fine-tuning and merging ViT-B/16 on eight vision tasks, and Tab. 14 shows the corresponding results for ViT-L/14. It can be observed that our DO-Merging method demonstrates consistently improved performance across both architectures. These results serve as supplementary validation for the superior performance of DO-Merging presented in the main text.

G Broader Impacts

This paper presents work whose goal is to advance the field of Model Merging for the efficient utilization and deployment of deep learning models. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.