GLOBA: Rethinking Parameter Conflicts in Model Merging

Zehao Liu[1,2], **Kun Li**[1*], **Wei Zhou**[1*]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{liuzehao, likun2, zhouwei}@iie.ac.cn

Abstract

Model merging serves as a training-free technique that combines multiple task-specific models into a unified multi-task model, but parameter conflicts often lead to performance drops. Previous methods flatten weight matrices into one-dimensional vectors, losing the inherent structural information of their row and column spaces. We mathematically prove and experimentally validate that parameter conflicts arise from non-orthogonal components of task vectors, while orthogonal components are conflict-free. Furthermore, we find that non-orthogonal components can contain both harmful conflicts and beneficial synergies. To precisely locate parameter conflicts and extract orthogonal components, we propose GLOBA (GLObal Basis Analysis Framework), which projects task vectors onto a global basis to align them within a unified coordinate system and construct a task interaction matrix. Following energy-based pruning, we divide parameters into five types based on the orthogonal relationships between the row spaces and column spaces of task vectors. Experiments on three fine-tuned models (mathematics, coding, and instruction-following) using LLaMA-2-7B and LLaMA-2-13B demonstrate significant performance gains through selective retention of beneficial parameters and removal of conflicting ones.

Code — https://github.com/Zehao-Liu/GLOBA

1 Introduction

Fine-tuning pre-trained large language models (LLMs) on downstream tasks has become the main approach for achieving better performance on specific tasks. This success has led to many task-specific models, but maintaining separate models for each task creates high storage and deployment costs while preventing knowledge sharing across related tasks. Model merging addresses these challenges by combining multiple expert models into a single multi-task model without additional training. Most existing methods merge models by directly combining their parameters.

Current model merging methods face a key challenge: parameter conflicts often cause performance drops. Task vectors, which encode the information necessary to perform well on a given task (Ilharco et al. 2023), represent the parameter differences between fine-tuned and pre-trained models. Existing work commonly assumes that parameter conflicts arise when multiple tasks modify parameters at the same positions (Yadav et al. 2023). However, these approaches treat weight matrices as flattened one-dimensional vectors, losing crucial geometric structure information. Specifically, changing a single parameter affects both the corresponding row and column vector, thereby changing the entire row and column space of the matrix.

We take a geometric view of parameter conflicts. Task vectors can be split into orthogonal and non-orthogonal components. We mathematically prove and experimentally validate that orthogonal components do not cause parameter conflicts, while parameter conflicts only arise from non-orthogonal components. Furthermore, non-orthogonal components are not always harmful. Some components hurt performance while others help. We call these non-orthogonal components "parameter overlap." Based on the orthogonal relationships between row spaces and column spaces of task vectors, we divide overlapping parameters into five types. To extract completely orthogonal components and find the harmful overlaps, we develop GLOBA (GLObal Basis Analysis Framework).

GLOBA addresses this by constructing a global basis, which is a set of basis vectors that can represent all task vectors to be merged without information loss. We obtain this global basis by first performing SVD on each individual task vector, then concatenating all singular vectors, and finally applying SVD to the concatenated singular vectors. This global basis then defines a unified coordinate system, which we call the global coordinate system. Through this approach, we align all tasks to common basis directions. GLOBA then projects each task vector onto this global coordinate system to form task interaction matrices. Each element quantifies how strongly a task modifies parameters in specific global basis directions. We apply energy-based pruning to focus on the most significant parameter modifications. This retains important interactions while filtering out noise. This unified representation enables us to systematically analyze overlap patterns. We can distinguish which overlapping parameters create harmful conflicts and which provide beneficial synergies. Through this approach, we se-

*Corresponding author.

lectively retain beneficial parameters while removing conflicting ones, leading to more effective model merging.

Our experiments demonstrate that GLOBA generally outperforms existing baseline methods including Task Arithmetic (Ilharco et al. 2023) and TIES-Merging (Yadav et al. 2023) by selectively retaining beneficial parameters while removing harmful ones. As shown in Tables 2 and 1, when merging 13B Math & Code models on MBPP, our method achieves 21.80% compared to Task Arithmetic's 8.80% (13.00% absolute improvement). When merging 7B Instruction & Code models on MBPP, our method reaches 31.60%, surpassing even the original 7B code model's 20.80% performance (10.80% absolute improvement). Critically, we discover that completely orthogonal parameters enable conflict-free merging, where most performance changes fall within ±1%. Meanwhile, overlapping parameters exhibit a dual nature: some create destructive conflicts that eliminate task capabilities, while others provide beneficial synergies that enhance performance beyond individual task models.

The main contributions of this paper are:

(1) Through row space and column space analysis, we mathematically prove and experimentally validate that orthogonal components of task vectors do not cause parameter conflicts, while conflicts only arise from non-orthogonal components.

(2) We find that non-orthogonal components are not always harmful. Some create conflicts while others may provide strong beneficial synergies.

(3) We propose GLOBA (GLObal Basis Analysis Framework), which projects task vectors onto a global basis to align them within a unified coordinate system. This enables us to extract completely orthogonal components and precisely locate parameter conflicts.

2 Related Work

Model merging originated from early ensemble methods such as Weight Averaging (Izmailov et al. 2018) and Model Soups (Wortsman et al. 2022), which demonstrated that linear interpolation of model parameters can achieve superior performance. The breakthrough came from Task Arithmetic (Ilharco et al. 2023), which introduced task vectors, defined as parameter differences between fine-tuned and pre-trained models ($\tau_i = \theta_{ft_i} - \theta_{pre}$), transforming multi-task fusion into simple vector arithmetic. However, direct task vector averaging often causes performance degradation due to parameter conflicts.

Subsequent work focused on addressing these conflicts through various strategies. TIES-Merging (Yadav et al. 2023) systematically removes parameters with inconsistent signs to eliminate sign divergence, while DARE (Yu et al. 2024) reduces interference through random dropout of conflicting parameters. Other approaches explore geometric alignment (Ainsworth, Hayase, and Srinivasa 2023; Jordan et al. 2023; Stoica et al. 2024), adaptive weighting strategies (Yang et al. 2024; Davari and Belilovsky 2024), importance-based fusion (Matena and Raffel 2022; Jin et al. 2023), and matrix decomposition techniques (Chaichana et al. 2025; Lu et al. 2024; Tang et al. 2024). Task Singular Vectors (Gargiulo et al. 2025) performs SVD decomposition on layer-wise task matrices and uses whitening transformations to orthogonalize singular vectors from different tasks. In contrast, we analyze parameter conflicts from the geometric perspective of row space and column space relationships, mathematically proving that orthogonal task vector components are conflict-free and classifying parameter overlaps into five distinct types to distinguish between conflicting and synergistic effects.

3 Background and Motivation

3.1 Background

Given pre-trained model parameters θ_{pre} and task-specific fine-tuned parameters $\{\theta_{ft_i}\}_{i=1}^{T}$, we define the task vector as the parameter difference between the fine-tuned and pre-trained models:

$$\tau_i = \theta_{ft_i} - \theta_{\text{pre}}. \tag{1}$$

Traditional model merging approaches achieve multi-task integration through direct summation of task vectors with a global scaling factor:

$$\theta_{\text{merged}} = \theta_{\text{pre}} + \alpha \sum_{i=1}^{T} \tau_i, \tag{2}$$

where α represents the scaling factor. However, this direct summation operation fundamentally neglects the parameter space relationships among task vectors.

To illustrate this limitation, consider the case of two-task merging. When task vectors are decomposed as $\tau_i = U_i \Sigma_i V_i^T$, the traditional summation operation becomes:

$$\tau_1 + \tau_2 = U_1 \Sigma_1 V_1^T + U_2 \Sigma_2 V_2^T. \tag{3}$$

This summation can be rewritten in block matrix form:

$$\tau_1 + \tau_2 = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}. \tag{4}$$

This reformulation reveals that traditional summation is mathematically equivalent to concatenating the singular vectors from different tasks. Singular vectors U_i and V_i represent the row space (input space) and column space (output space) of each task vector respectively. Parameter conflicts arise when concatenating matrices $[U_1 \quad U_2]$ and $[V_1 \quad V_2]$ if the row spaces or column spaces are non-orthogonal ($U_1^T U_2 \neq 0$ or $V_1^T V_2 \neq 0$). When both spaces are completely orthogonal ($U_1^T U_2 = 0$ and $V_1^T V_2 = 0$), tasks operate in separate subspaces without conflicts.

3.2 Parameter Space Overlap in Model Merging

To understand parameter conflicts geometrically, we analyze task vectors from the perspective of input-output space transformations. For any task vector $\tau \in \mathbb{R}^{m \times n}$ acting as a linear transformation, it defines a mapping from an n-dimensional input space to an m-dimensional output space.

When an input vector $\mathbf{x} \in \mathbb{R}^n$ undergoes transformation through task vector τ, the operation $\mathbf{y} = \tau\mathbf{x}$ can be characterized through SVD decomposition $\tau = U\Sigma V^T$:

$$\mathbf{y} = U\Sigma V^T \mathbf{x}. \tag{5}$$

This reveals that: (1) the row space spanned by V determines which directions in the input space can be perceived and processed by the task vector; (2) the column space spanned by U determines the output directions where the task vector can generate responses.

The orthogonal relationships between row and column spaces of different task vectors lead to four distinct cases:

Row space overlap ($V_1^T V_2 \neq 0, U_1^T U_2 = 0$): Both tasks modify the same input directions, causing interference in input processing.

Column space overlap ($U_1^T U_2 \neq 0, V_1^T V_2 = 0$): Both tasks generate outputs in the same directions, causing interference in output generation.

Both spaces overlap ($V_1^T V_2 \neq 0, U_1^T U_2 \neq 0$): Both input and output spaces overlap, causing interference in both input processing and output generation.

Complete orthogonality ($V_1^T V_2 = 0, U_1^T U_2 = 0$): Tasks operate in disjoint subspaces, ensuring no interference.

The complete orthogonality case motivates us to mathematically prove that orthogonal components are conflict-free. We also need to develop a framework for extracting orthogonal components from task vectors and exploring the effects of different parameter overlap patterns.

4 Methodology

4.1 Orthogonal Components are Conflict-Free

We establish that orthogonal components of task vectors can be merged without conflicts by analyzing their fundamental subspace relationships.

For each layer, we define task vector $\tau \in \mathbb{R}^{m \times n}$ as the parameter difference matrix. Since most layers perform linear transformations, each layer's task vector can be decomposed into components that are completely orthogonal to other task vectors and components that have non-orthogonal overlaps. We focus our analysis on these orthogonal components.

Given two task vectors $\tau_A, \tau_B \in \mathbb{R}^{m \times n}$ from each layer, we can decompose τ_A as:

$$\tau_A = \tau_A^{\perp} + \tau_A^{\text{overlap}}, \tag{6}$$

where $\tau_A^{\perp}$ is the component of τ_A that is completely orthogonal to τ_B, and τ_A^{overlap} contains the overlapping parts.

Proposition. *Orthogonal components ensure conflict-free merging.* Given the orthogonal component $\tau_A^{\perp}$ that satisfies:

$$\text{Row}(\tau_A^{\perp}) \perp \text{Row}(\tau_B) \quad \text{and} \quad \text{Col}(\tau_A^{\perp}) \perp \text{Col}(\tau_B), \tag{7}$$

then merging $\tau_{\text{merged}} = \tau_A^{\perp} + \tau_B$ preserves each task's contribution without interference.

Proof:

1. For any input vector $\mathbf{x} \in \mathbb{R}^n$, since the row spaces are orthogonal, we can decompose it as $\mathbf{x} = \mathbf{x}_A^{\perp} + \mathbf{x}_B + \mathbf{x}_{\text{null}}$, where $\mathbf{x}_A^{\perp} \in \text{Row}(\tau_A^{\perp})$, $\mathbf{x}_B \in \text{Row}(\tau_B)$, and $\mathbf{x}_{\text{null}}$ is orthogonal to both row spaces.
2. We analyze the action of $\tau_A^{\perp}$ on $\mathbf{x}$:
$$\tau_A^{\perp}(\mathbf{x}) = \tau_A^{\perp}(\mathbf{x}_A^{\perp}) + \tau_A^{\perp}(\mathbf{x}_B) + \tau_A^{\perp}(\mathbf{x}_{\text{null}}). \tag{8}$$
Since $\text{Row}(\tau_A^{\perp}) \perp \text{Row}(\tau_B)$, we have $\mathbf{x}_B \in \text{Null}(\tau_A^{\perp})$, thus $\tau_A^{\perp}(\mathbf{x}_B) = 0$. Similarly, $\tau_A^{\perp}(\mathbf{x}_{\text{null}}) = 0$. Therefore, $\tau_A^{\perp}(\mathbf{x}) = \tau_A^{\perp}(\mathbf{x}_A^{\perp}) = \mathbf{y}_A^{\perp}$.
3. Similarly, $\tau_B(\mathbf{x}) = \tau_B(\mathbf{x}_B) = \mathbf{y}_B$.
4. The merged output is:
$$\tau_{\text{merged}}(\mathbf{x}) = \tau_A^{\perp}(\mathbf{x}) + \tau_B(\mathbf{x}) = \mathbf{y}_A^{\perp} + \mathbf{y}_B. \tag{9}$$
5. Since $\text{Col}(\tau_A^{\perp}) \perp \text{Col}(\tau_B)$, the output vectors $\mathbf{y}_A^{\perp}$ and $\mathbf{y}_B$ are orthogonal and cannot cancel each other out.

Each task's contribution is completely preserved without interference. The detailed proof for decomposing each layer's task vectors into orthogonal and non-orthogonal components is provided in the appendix.

This result reveals that parameter conflicts arise from non-orthogonal components, while orthogonal components can be safely merged with any other task vectors.

4.2 GLOBA: GLObal Basis Analysis Framework

Overview. Based on our theoretical analysis in Section 4.1, we proved that orthogonal components of task vectors do not cause parameter conflicts, while conflicts only arise from non-orthogonal components. This raises two critical questions: First, can we extract completely orthogonal components from task vectors and validate experimentally whether they truly enable conflict-free merging? Second, to understand where parameter conflicts actually occur within overlapping parameters, can we systematically classify overlapping parameters based on their overlap patterns and explore the specific effects of different overlap types? These questions motivate us to develop a framework that can precisely extract orthogonal components and systematically categorize parameter overlaps for empirical investigation.

To address these challenges, we propose GLOBA (GLObal Basis Analysis Framework), which constructs a global basis that can represent all task vectors without information loss. By projecting task vectors onto this unified coordinate system, GLOBA aligns them to common basis directions, enabling us to extract completely orthogonal components and classify overlapping parameters into five distinct types based on their geometric relationships in row and column spaces. After energy-based pruning that focuses on significant parameter interactions, we can then experimentally validate whether orthogonal components are truly conflict-free and explore how different overlap patterns affect merging performance, leading to adaptive strategies for handling different situations.

Global Basis Construction. To extract orthogonal components and analyze parameter overlaps, we construct a global basis that spans the union of all task vector spaces, providing a unified coordinate system for analysis. We first perform singular value decomposition on each task vector:

$$\tau_i = U_i \Sigma_i V_i^T, \quad i = 1, 2, \ldots, T. \tag{10}$$

Subsequently, we concatenate the left and right singular vectors from all tasks:

$$U_{\text{concat}} = [U_1 \quad U_2 \quad \cdots \quad U_T], \tag{11}$$

$$V_{\text{concat}} = [V_1 \quad V_2 \quad \cdots \quad V_T]. \tag{12}$$

We then perform SVD decomposition on the concatenated singular vectors to obtain the global bases:

$$U_{\text{concat}} = U_{\text{global}}\Sigma^{(U)}_{\text{global}}V^{(U)T}_{\text{global}}, \tag{13}$$

$$V_{\text{concat}} = V_{\text{global}}\Sigma^{(V)}_{\text{global}}V^{(V)T}_{\text{global}}. \tag{14}$$

U_{global} and V_{global} no longer belong to any single task, but rather constitute a unified coordinate system for all tasks to be merged. This global basis captures the principal variation directions of all task vectors, establishing the foundation for subsequent unified analysis.

Task Interaction Matrix Construction and Energy-based Pruning. Having obtained the global basis, we project each task vector onto this unified coordinate system to obtain the task interaction matrix:

$$\hat{C}_i = U^T_{\text{global}}\tau_i V_{\text{global}}. \tag{15}$$

The projection coefficient $\hat{C}_i[r, c]$ possesses clear geometric meaning, representing the projection intensity of task vector τ_i in the global basis direction (u_r, v_c):

$$\hat{C}_i[r, c] = \langle \tau_i, u_r v_c^T \rangle. \tag{16}$$

To focus on the most important interaction patterns, we employ an energy-based pruning strategy. We rank all positions by their squared magnitudes and retain only those positions whose cumulative energy constitutes the top η fraction of the total energy:

$$\tilde{C}_i[r, c] = \begin{cases} \hat{C}_i[r, c] & \text{if } (r, c) \in \text{top } \eta \text{ energy positions} \\ 0 & \text{otherwise} \end{cases}, \tag{17}$$

where energy is measured as $|\hat{C}_i[r, c]|^2$. Detailed settings for η across different experiments are provided in the appendix.

Unlike traditional methods that preserve the largest singular values within individual task coordinate systems, our global energy-based pruning ensures that all tasks focus on the same globally significant interaction directions, creating a unified framework for meaningful cross-task analysis while filtering out task-specific noise that may interfere with the merging process.

Parameter Overlap Classification and Analysis. The task interaction matrix $\tilde{C}_i$ provides a structured representation where rows correspond to global left singular vector directions (output space basis) and columns correspond to global right singular vector directions (input space basis). Each element $\tilde{C}_i[r, c]$ quantifies the interaction strength between specific input-output transformation directions for task i within the unified global coordinate system.

Our framework divides all possible parameter relationships into five comprehensive types that directly correspond to the four fundamental cases identified in our geometric analysis of parameter space overlap. **Type A (Complete Orthogonality)** represents the ideal case where tasks operate in entirely separate input-output subspaces with no parameter conflicts, ensuring interference-free merging as proven in our theoretical analysis. **Type B (Pure Row Orthogonality)** occurs when tasks maintain orthogonal row spaces while sharing overlapping column spaces. Tasks process different input directions but generate outputs that affect similar output dimensions, creating parameter space overlap only in the output generation aspects. **Type C (Pure Column Orthogonality)** represents the complementary scenario where tasks maintain orthogonal column spaces while sharing overlapping row spaces. Tasks respond to similar input directions but generate outputs in different output dimensions, creating parameter space overlap only in the input processing aspects. **Type D (Direct Overlap)** encompasses scenarios where tasks modify parameters in exactly the same input-output transformation directions within the global basis. This subdivides into **Type D$^+$ (Same-sign Overlap)** where tasks contribute modifications in the same direction at identical positions, creating amplified effects, and **Type D$^-$ (Opposite-sign Overlap)** where tasks contribute modifications in opposite directions at identical positions, creating cancellation effects. **Type E (Structural Overlap)** identifies cases where both row spaces and column spaces overlap but tasks do not directly overlap at specific matrix positions. For example, if one task modifies position (r,c1) and another task modifies position (r,c2), both row and column spaces are overlapping but the tasks affect different specific positions within the shared subspaces.

This comprehensive classification enables us to distinguish between harmful conflicts and beneficial synergies within overlapping parameters, rather than treating all overlapping parameters as conflicts. As demonstrated in our experiments, this fine-grained understanding enables adaptive merging strategies. These strategies selectively retain beneficial interactions while mitigating harmful conflicts, leading to more effective model merging. The appendix provides the specific mathematical formulations for identifying these parameter types within the task interaction matrix, as well as their correspondence to the geometric overlap patterns.

Improved Merging Strategies. Based on our systematic analysis of different overlap types in parameter space, we propose three improved merging strategies that can selectively handle different overlap patterns:

Strategy 1: Uniform Scaling

$$\tilde{C}_{\text{merged}} = \alpha \cdot (\tilde{C}_1 + \tilde{C}_2). \tag{18}$$

Strategy 2: Selective Inclusion

$$\tilde{C}_{\text{merged}}[r, c] = \begin{cases} \tilde{C}_1[r, c] + \tilde{C}_2[r, c] & \text{if } (r, c) \in S \\ 0 & \text{otherwise} \end{cases}, \tag{19}$$

where S represents the set of positions belonging to selected overlap types.

Strategy 3: Type-specific Weighting

$$\tilde{C}_{\text{merged}}[r, c] = \alpha_{\text{type}(r,c)} \cdot (\tilde{C}_1[r, c] + \tilde{C}_2[r, c]), \tag{20}$$

where $\alpha_{\text{type}(r,c)}$ represents the type-specific scaling factor determined empirically for each overlap type.

After obtaining the improved task interaction matrix, we reconstruct the final merged task vector:

$$\tau_{\text{merged}} = U_{\text{global}}\tilde{C}_{\text{merged}}V^T_{\text{global}}. \tag{21}$$

Models	Methods	Instruction following	Math Reasoning	Code Generation
		AlpacaEval	GSM8K	MBPP
Instruct Model	/	31.66	/	/
Math Model	/	/	55.57	/
Code Model	/	/	/	20.80
Instruct Model & Math Model	TA	35.11	48.75	/
	TIES	**39.93**	45.87	/
	WA	32.90	46.25	/
	DARE	33.75	48.98	/
	Ours	39.14	**49.96**	/
Instruct Model & Code Model	TA	**33.95**	/	14.00
	TIES	19.52	/	18.60
	WA	32.59	/	7.20
	DARE	/	/	/
	Ours	29.04	/	**31.60**
Math Model & Code Model	TA	/	51.33	15.00
	TIES	/	4.32	**22.40**
	WA	/	45.87	13.80
	DARE	/	/	/
	Ours	/	**51.48**	18.80

Table 1: Performance of merging Llama-2-7b-instruct, WizardMath-7B-V1.0, and llama2_7b_code using different merging methods.

Models	Methods	Instruction following	Math Reasoning	Code Generation
		AlpacaEval	GSM8K	MBPP
Instruct Model	/	77.48	/	31.20
Math Model	/	45.47	58.45	/
Code Model	/	37.02	/	28.00
Instruct Model & Math Model	TA	77.03	60.96	/
	TIES	**77.90**	27.60	/
	WA	75.63	63.08	/
	DARE	75.03	61.64	/
	Ours	77.58	**63.61**	/
Instruct Model & Code Model	TA	77.97	/	32.40
	TIES	75.25	/	31.20
	WA	75.87	/	32.00
	DARE	76.93	/	32.20
	Ours	**82.12**	/	**36.80**
Math Model & Code Model	TA	/	58.91	8.80
	TIES	/	**59.97**	15.60
	WA	/	50.49	8.40
	DARE	/	59.14	8.60
	Ours	/	59.67	**21.80**

Table 2: Performance of merging WizardLM-13B-v1.2, WizardMath-13B-v1.0, and llama-2-13b-code-alpaca using different merging methods.

Theoretical Foundation. To establish the theoretical foundation of our approach, we demonstrate the mathematical equivalence between task vector merging and task interaction matrix operations.

For the unpruned case, consider any task vector τ_i and its projection:

$$\hat{C}_i = U_{\text{global}}^T \tau_i V_{\text{global}}. \tag{22}$$

This can be reconstructed as $\tau_i = U_{\text{global}} \hat{C}_i V_{\text{global}}^T$. When merging two task vectors:

$$\tau_1 + \tau_2 = U_{\text{global}} (\hat{C}_1 + \hat{C}_2) V_{\text{global}}^T. \tag{23}$$

This equivalence holds because U_{global} spans the column spaces of all task vectors and V_{global} spans the row spaces of all task vectors, ensuring that any linear combination can be perfectly represented through corresponding matrix operations.

5 Experiments

5.1 Experimental Setup

Models and Datasets. We conduct experiments on three task-specific fine-tuned models: instruction-following, mathematical reasoning, and code generation. For instruction-following, we use Llama-2-7b-instruct (circulus 2023) and WizardLM-13B-V1.0 (Xu et al. 2023), evaluated on AlpacaEval (Li et al. 2023). For mathematical reasoning, we employ WizardMath-7B-V1.0 and WizardMath-13B-V1.O (Luo et al. 2023), tested on GSM8K (Cobbe et al. 2021). For code generation, we use llama2_7b_code (itsliupeng 2023) and llama-2-13b-code-alpaca (Chaudhary 2023), evaluated on MBPP (Austin et al. 2021). All models are fine-tuned based on LLaMA-2 (Touvron et al. 2023) pre-trained backbones.

Evaluation Metrics. We use win rate for AlpacaEval, zero-shot accuracy for GSM8K, and pass@1 accuracy for MBPP to measure task-specific performance across different domains.

Baselines. We compare against four established model merging methods: Task Arithmetic (TA) (Ilharco et al. 2023), TIES-Merging (TIES) (Yadav et al. 2023), Weight Averaging (WA) (Izmailov et al. 2018), and DARE (Yu et al. 2024). These methods represent the leading-edge approaches in model merging.

5.2 Main Results

Table 1 and Table 2 present comprehensive comparison results for 7B and 13B model merging across three task domains. Our GLOBA framework consistently achieves superior performance compared to existing baselines.

For 7B models, merging math and code models, GLOBA achieves 18.80% on MBPP, representing a 3.80% absolute improvement over Task Arithmetic's 15.00%. More remarkably, merging instruction and code models, our method achieves 31.60% on MBPP—surpassing even the original standalone code model's 20.80% performance by 10.80% absolute improvement. Note that DARE (Yu et al. 2024) performs poorly on 7B code fine-tuned models due to the large absolute values in the delta matrices, making random dropout destructive to important parameter modifications.

For 13B models, when merging math and code models, GLOBA achieves 21.80% on MBPP—a substantial 13.00% absolute improvement over Task Arithmetic's 8.80% and 6.20% absolute improvement over TIES-Merging's 15.60%.

Model Pair	Before Orth.	After Orth.	Change (%)
7B Models			
Math + Instruct Orth.	49.20	50.27	+2.2%
Instruct + Code Orth.	45.96	45.49	-1.0%
Math + Code Orth.	51.63	51.48	-0.3%
13B Models			
Math + Instruct Orth.	58.98	59.21	+0.4%
Instruct + Code Orth.	81.29	82.12	+1.0%
Math + Code Orth.	59.29	59.06	-0.4%

Table 3: Performance impact of adding orthogonal patterns after energy-based pruning. Minimal changes (within $\pm 2.2\%$) validate conflict-free merging of orthogonal components.

Configuration	Math (GSM8K)	Code (MBPP)
WizardMath-7B Only	51.63	/
llama2_7b_code Only	/	22.80
+ Type A (Complete Orth.)	**51.48**	**18.80**
+ Type B (Row Orth.)	43.06	16.00
+ Type C (Column Orth.)	47.46	10.40
+ Type E (Structural)	48.29	15.40
+ Type D^+ (Same-sign)	5.38	0
+ Type D^- (Opposite)	0	12.80
+ All Patterns	43.14	8.80

Table 4: Math-code model merging with energy-based pruning ($\eta = 0.95$ for WizardMath-7B-V1.0, $\eta = 0.8$ for llama2_7b_code). Each row adds different overlap patterns from llama2_7b_code to WizardMath-7B-V1.0 baseline. The final row shows combined effect of all patterns. Type D patterns cause severe performance drops.

When merging instruction and code models, GLOBA achieves 82.12% on AlpacaEval and 36.80% on MBPP, outperforming all baseline methods.

5.3 Discussion

RQ1: Do Orthogonal Components Truly Enable Conflict-Free Merging as Theoretically Predicted? To experimentally validate our first theoretical contribution that orthogonal components never cause parameter conflicts, we analyze the performance impact of adding completely orthogonal patterns from auxiliary models to base models after energy-based pruning.

Table 3 provides compelling empirical evidence supporting our mathematical proof. Across all model pairs and sizes, the performance changes when incorporating Type A (Complete Orthogonality) patterns remain remarkably minimal: the largest change is +2.2% for Math + Instruct merging in 7B models, while most changes fall within $\pm 1\%$. This experimental validation confirms that orthogonal components can be safely merged without introducing harmful conflicts, as they operate in entirely independent parameter subspaces.

Crucially, this stability serves as a theoretical foundation for understanding parameter conflicts. The near-zero performance impact of orthogonal patterns establishes a baseline that allows us to attribute all significant performance changes in model merging to non-orthogonal interactions, providing empirical evidence that parameter conflicts arise exclusively from overlapping parameter subspaces rather than orthogonal ones.

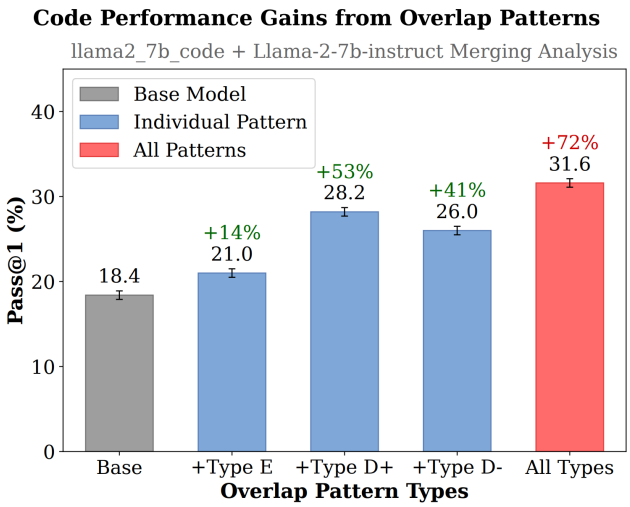

Figure 1: Code performance gains from merging llama2_7b_code and Llama-2-7b-instruct (Pass@1 on MBPP). Each overlap pattern provides consistent gains, achieving +72% improvement when all patterns are combined. Only Type E, Type D+, and Type D- patterns exist after energy-based pruning.

RQ2: How Do Different Task Combinations Exhibit Distinct Overlap Pattern Characteristics? Different task combinations exhibit three distinct overlap pattern scenarios with dramatically different performance implications. Table 4 shows a conflict-dominated scenario in WizardMath-7B and llama2_7b_code merging, where Type D+ (Same-sign) overlaps cause catastrophic performance drop (mathematical reasoning: 51.63% $\rightarrow$ 5.38%) and Type D- (Opposite-sign) patterns completely eliminate capabilities (0% math performance). This suggests that math and code reasoning operate in fundamentally incompatible parameter subspaces for certain transformation directions.

Figure 1 shows a synergy-dominated scenario in code-instruction merging, where all overlap types (E, D^+, D^-) contribute positively, achieving +72% improvement through beneficial complementary relationships. Table 5 presents a balanced scenario with mixed beneficial (A, B, C, E, D^+) and conflicting (D^-) patterns, requiring selective management strategies.

RQ3: Why Does Energy-Based Pruning Enable Effective Parameter Conflict Resolution? Energy-based pruning with threshold η filters task-specific noise that obscures beneficial parameter relationships, focusing only on globally significant interaction directions. Figure 1 reveals the transformative power of this approach. While traditional Task

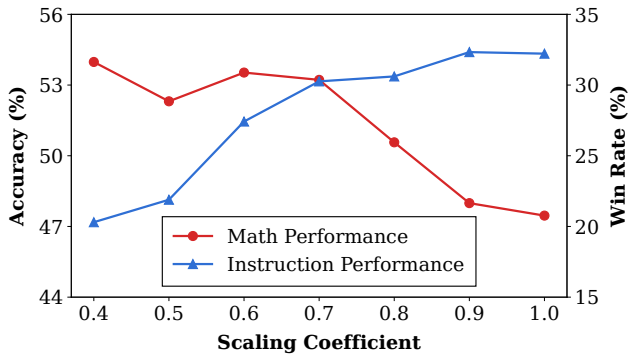

Figure 2: Selective pattern merging with type-specific weighting when adding Llama-2-7b-instruct to WizardMath-7B-V1.0 after removing Type D$^-$ patterns. Scaling remaining beneficial patterns by coefficient α shows trade-off: math performance improves (47.46% to 53.98%) while instruction performance declines as α decreases.

Configuration	Math (GSM8K)	Instruction (AlpacaEval)
WizardMath-7B Only	49.20	13.00
Llama-2-7b-instruct Only	0	41.88
+ Type A (Complete Orth.)	50.27	13.40
+ Type B (Row Orth.)	**52.16**	18.67
+ Type C (Column Orth.)	51.33	17.09
+ Type E (Structural)	51.10	22.45
+ Type D$^+$ (Same-sign)	52.01	16.57
+ Type D$^-$ (Opposite)	44.43	16.26
+ All Patterns	45.56	**38.57**

Table 5: Math-instruction model merging with energy-based pruning ($\eta = 0.8$ for both models). Each row adds different overlap patterns from Llama-2-7b-instruct to WizardMath-7B-V1.0 baseline. The final row shows combined effect of all patterns. Results show mixed beneficial and conflicting patterns.

Arithmetic shows apparent conflicts (20.80% $\rightarrow$ 14.00% on MBPP), our energy-based pruning uncovers that all overlap patterns (Type E, D+, and D-) actually provide consistent performance gains when noise is removed.

The dramatic improvement (+72%) shows that what appears as destructive interference often contains hidden complementary relationships that conventional methods miss. Energy-based pruning enables our global coordinate system to distinguish genuine conflicts from beneficial synergies masked by irrelevant modifications, revealing the underlying geometric structure governing successful model merging.

RQ4: How Do Type-Specific Weighting Strategies Achieve Optimal Balance? Our type-specific weighting enables fine-grained control over parameter interactions through principled coefficient optimization. Figure 2 demonstrates the fundamental trade-off in multi-task merging: after removing harmful Type D$^-$ patterns and scaling beneficial patterns by coefficient α, mathematical performance improves from 47.46% to 53.98% as α decreases from 1.0 to 0.4, while instruction performance declines from 32.79% to 25.26%. This reveals that excessive parameter modifications can cause models to deviate from optimal task-specific manifolds (Ortiz-Jimenez, Favero, and Frossard 2023), necessitating careful scaling to balance competing objectives rather than simply maximizing beneficial interactions.

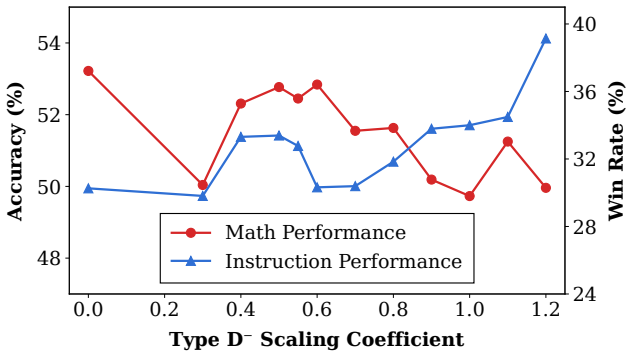

Figure 3: Fine-grained Type D$^-$ scaling when adding Llama-2-7b-instruct to WizardMath-7B-V1.0 with beneficial patterns fixed at 0.7×. Varying Type D$^-$ coefficient from 0 to 1.2 shows optimal balance at 1.2, improving instruction performance (30.26%→39.14%) while maintaining math performance at 49.96%.

Figure 3 shows our solution: fixing beneficial patterns at 0.7$\times$ scaling while varying Type D$^-$ coefficients achieves optimal balance at coefficient 1.2, reaching 39.14% instruction performance and 49.96% mathematical performance. This demonstrates that strategic retention of seemingly conflicting patterns can enhance overall capability when properly calibrated, significantly outperforming simple removal strategies that achieve only 30.26% instruction and 52.16% mathematical performance.

6 Conclusion

This work analyzes parameter conflicts in model merging from a geometric perspective by mathematically proving that conflicts arise exclusively from non-orthogonal components of task vectors, while orthogonal components enable conflict-free merging. We find that non-orthogonal components exhibit a dual nature—some create destructive conflicts while others provide beneficial synergies. We propose GLOBA (GLObal Basis Analysis Framework), which projects task vectors onto a global basis and constructs a task interaction matrix, enabling precise extraction of orthogonal components and classification of overlapping parameters into five distinct types. Through energy-based pruning and type-specific weighting strategies, GLOBA outperforms existing baselines across mathematics, coding, and instruction-following tasks. Our approach moves beyond simple conflict avoidance to actively leverage beneficial parameter interactions, establishing a geometric framework for understanding parameter conflicts in model merging.

Acknowledgments

This work is supported by Youth Innovation Promotion Association CAS. We thank the shepherd and all the anonymous reviewers for their constructive feedback.

References

Ainsworth, S.; Hayase, J.; and Srinivasa, S. 2023. Git rebasin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations*.

Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C. J.; Terry, M.; Le, Q. V.; and Sutton, C. 2021. Program Synthesis with Large Language Models. arXiv:2108.07732.

Chaichana, Y.; Trachu, T.; Limkonchotiwat, P.; et al. 2025. Decom-Renorm-Merge: Model Merging on the Right Space Improves Multitasking. arXiv:2505.23117.

Chaudhary, S. 2023. Code Alpaca: An Instruction-following LLaMA model for code generation. https://github.com/sahil280114/codealpaca. Accessed: 2025-07-15.

circulus. 2023. circulus/Llama-2-7b-instruct. https://huggingface.co/circulus/Llama-2-7b-instruct. Accessed: 2025-06-20.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168.

Davari, M.; and Belilovsky, E. 2024. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *European Conference on Computer Vision*.

Gargiulo, A. A.; Crisostomi, D.; Bucarelli, M. S.; et al. 2025. Task Singular Vectors: Reducing Task Interference in Model Merging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ilharco, G.; Ribeiro, M. T.; Wortsman, M.; et al. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

itsliupeng. 2023. itsliupeng/llama2_7b_code. https://huggingface.co/itsliupeng/llama2_7b_code. Accessed: 2025-07-22.

Izmailov, P.; Podoprikhin, D.; Garipov, T.; et al. 2018. Averaging weights leads to wider optima and better generalization. arXiv:1803.05407.

Jin, X.; Ren, X.; Preotiuc-Pietro, D.; and Cheng, P. 2023. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*.

Jordan, K.; Sedghi, H.; Saukh, O.; et al. 2023. REPAIR: REnormalizing permuted activations for interpolation repair. In *The Eleventh International Conference on Learning Representations*.

Li, X.; Zhang, T.; Dubois, Y.; Taori, R.; Gulrajani, I.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. AlpacaEval: An Automatic Evaluator of Instruction-following Models. https://github.com/tatsu-lab/alpaca_eval. Accessed: 2025-06-20.

Lu, Z.; Fan, C.; Wei, W.; Qu, X.; Chen, D.; and Cheng, Y. 2024. Twin-Merging: Dynamic Integration of Modular Expertise in Model Merging. In *Advances in Neural Information Processing Systems*.

Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; and Zhang, D. 2023. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. arXiv:2308.09583.

Matena, M. S.; and Raffel, C. A. 2022. Merging models with fisher-weighted averaging. In *Advances in Neural Information Processing Systems*, volume 35, 17703–17716.

Ortiz-Jimenez, G.; Favero, A.; and Frossard, P. 2023. Task arithmetic in the tangent space: improved editing of pre-trained models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*.

Stoica, G.; Bolya, D.; Bjorner, J. B.; et al. 2024. Zipit! merging models from different tasks without training. In *The Twelfth International Conference on Learning Representations*.

Tang, A.; Shen, L.; Luo, Y.; et al. 2024. Smile: Zero-shot sparse mixture of low-rank experts construction from pre-trained foundation models. arXiv:2408.10174.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; Bikel, D.; Blecher, L.; Ferrer, C. C.; Chen, M.; Cucurull, G.; Esiobu, D.; Fernandes, J.; Fu, J.; Fu, W.; Fuller, B.; Gao, C.; Goswami, V.; Goyal, N.; Hartshorn, A.; Hosseini, S.; Hou, R.; Inan, H.; Kardas, M.; Kerkez, V.; Khabsa, M.; Kloumann, I.; Korenev, A.; Koura, P. S.; Lachaux, M.-A.; Lavril, T.; Lee, J.; Liskovich, D.; Lu, Y.; Mao, Y.; Martinet, X.; Mihaylov, T.; Mishra, P.; Molybog, I.; Nie, Y.; Poulton, A.; Reizenstein, J.; Rungta, R.; Saladi, K.; Schelten, A.; Silva, R.; Smith, E. M.; Subramanian, R.; Tan, X. E.; Tang, B.; Taylor, R.; Williams, A.; Kuan, J. X.; Xu, P.; Yan, Z.; Zarov, I.; Zhang, Y.; Fan, A.; Kambadur, M.; Narang, S.; Rodriguez, A.; Stojnic, R.; Edunov, S.; and Scialom, T. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

Wortsman, M.; Ilharco, G.; Gadre, S. Y.; et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proceedings of the 39th International Conference on Machine Learning*.

Xu, C.; Sun, Q.; Zheng, K.; Geng, X.; Zhao, P.; Feng, J.; Tao, C.; and Jiang, D. 2023. WizardLM: Empowering Large Language Models to Follow Complex Instructions. arXiv:2304.12244.

Yadav, P.; Tam, D.; Choshen, L.; et al. 2023. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*.

Yang, E.; Wang, Z.; Shen, L.; et al. 2024. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*.

Yu, L.; Yu, B.; Yu, H.; et al. 2024. Language models are super mario: Absorbing abilities from homologous models

as a free lunch. In *Forty-first International Conference on Machine Learning*.