SPECTRAL COMPONENTS REQUIRE DISTINCT COEFFICIENTS IN MODEL MERGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Model merging combines task-specific experts into a single multi-task model without joint retraining. Cross-task orthogonalization eliminates overlap between the retained task subspaces in weight space, yet the merged model's predictions still differ from those of the task experts. We show that reweighting retained spectral components reduces this expert discrepancy while keeping their directions fixed. We find that spectral components from the same task and layer affect expert discrepancy differently and therefore require distinct coefficients. Accordingly, we jointly learn component-wise spectral coefficients in a fixed, task-structured orthogonal basis by matching expert predictions on unlabeled calibration inputs. Across three CLIP backbones and benchmarks of 8, 14, and 20 tasks, our method achieves the highest accuracy among the evaluated methods in all nine settings. On ViT-B/32, it improves normalized accuracy over the strongest baseline by 3.2, 4.0, and 6.3 percentage points with 8, 14, and 20 tasks, respectively. Experiments on MergeBench further show improvements over the evaluated merging baselines for both base and instruction-tuned Llama-3.2-3B models.

1 INTRODUCTION

Fine-tuning a shared pre-trained model on each downstream task produces task-specific experts, but increases storage and serving cost. Model merging combines these experts into a single model without joint retraining. Task Arithmetic (Ilharco et al., 2023) represents each expert as a task vector, defined as the weight difference between the expert and the pre-trained model, and sums these vectors in parameter space. However, adding task vectors from different tasks causes inter-task interference, which remains a central challenge in model merging.

Spectral merging methods modify the directions or magnitudes of spectral components in task-specific or merged weight updates to mitigate interference (Gargiulo et al., 2025; Marczak et al., 2025; Li et al., 2026b;c). TSV-M (Gargiulo et al., 2025) orthogonalizes the retained task subspaces across tasks. However, we observe that the merged model's predictions still differ from those of the corresponding task experts, a mismatch we refer to as expert discrepancy. As illustrated in Figure 1, reweighting the retained spectral components further reduces this discrepancy while keeping their directions fixed.

Existing adaptive merging methods learn coefficients at coarser granularities (Yang et al., 2024; Zhang et al., 2026), while AdaRank (Lee et al., 2026) adapts spectral components through binary selection. Our analysis reveals that a small increase in a component's coefficient can reduce expert discrepancy on its source task while increasing the aggregate discrepancy over the other tasks, or vice versa. These effects vary among components from the same task and layer. We therefore keep the task-structured orthogonal basis fixed and learn a separate coefficient for each retained component. All coefficients are optimized jointly to match the predictive distributions of the task experts on unlabeled calibration inputs.

Across three CLIP backbones (Radford et al., 2021) and benchmarks of 8, 14, and 20 tasks, our method achieves the highest accuracy among the evaluated methods in all nine settings. On ViT-B/32, it exceeds the best-performing baseline in each setting by 3.2, 4.0, and 6.3 percentage points in normalized accuracy with 8, 14, and 20 tasks, respectively. On MergeBench (He et al., 2025), it also achieves the highest mean domain score among the evaluated merging methods for both Llama-3.2-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

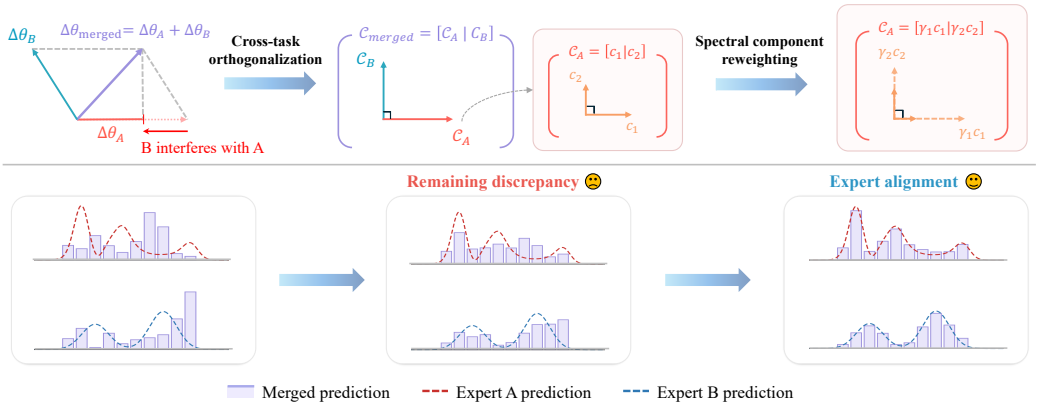


Figure 1: Cross-task orthogonalization eliminates task-subspace overlap but leaves expert discrepancy. Spectral component reweighting reduces the remaining discrepancy. Here, $\Delta\theta_t$ denotes the task vector for t, $\mathcal{C}_t$ its collection of retained spectral components after orthogonalization, c_j an individual component, and γ_j its learned scaling coefficient.

3B and Llama-3.2-3B-Instruct. Our ablations support learning component-wise coefficients jointly across tasks and layers with an expert-aligned objective.

Our contributions are threefold.

- We identify expert discrepancy that remains after cross-task orthogonalization and show that components originating from the same task and layer have heterogeneous effects on this discrepancy.
- We propose spectral component-wise reweighting in a fixed, task-structured orthogonal basis, jointly learning coefficients from expert predictions on unlabeled calibration inputs.
- We demonstrate the effectiveness of our method across three CLIP backbones with 8, 14, and 20 vision tasks, as well as two language-model settings, achieving the highest average performance among the evaluated merging methods in each setting.

2 Related Work

Model Merging. Model merging combines multiple fine-tuned models into a single model. Early approaches include weight averaging and Fisher-weighted merging (Wortsman et al., 2022; Matena & Raffel, 2022). Task Arithmetic (Ilharco et al., 2023) introduced task vectors, defined as the parameter differences between fine-tuned and pre-trained models, and showed that adding these vectors can combine capabilities from multiple tasks. Subsequent work reduces interference among task updates through sparsification and parameter-wise reconciliation (Yadav et al., 2023; Yu et al., 2024; Du et al., 2024).

Spectral Model Merging. A complementary line of work exploits the matrix and spectral structure of task updates rather than treating model parameters as flat vectors. TSV-M (Gargiulo et al., 2025) retains dominant singular components of layer-wise task matrices and orthogonalizes cross-task singular directions to reduce interference. Iso-CTS (Marczak et al., 2025) combines common and task-specific subspaces with an isotropic singular-value spectrum of the merged matrix. ESM (Li et al., 2026b) instead constructs essential subspaces from the principal directions of feature shifts induced by task updates and applies polarized scaling. SVC (Li et al., 2026c) identifies spectral over-accumulation of shared directions and rescales the singular values of an already merged update in a training- and data-free manner. All of these methods set the magnitude of each component by a fixed rule derived from the task updates or from the feature shifts they induce, rather than from agreement with the expert predictions.

Adaptive Model Merging. Another line of work learns merging coefficients from unlabeled data. AdaMerging (Yang et al., 2024), AdaRank (Lee et al., 2026), and SWIM-Merging (Li et al., 2026a) follow a test-time adaptation paradigm, using objectives derived from the merged model's own predictions, typically entropy minimization. They differ mainly in the adapted unit: AdaMerging learns

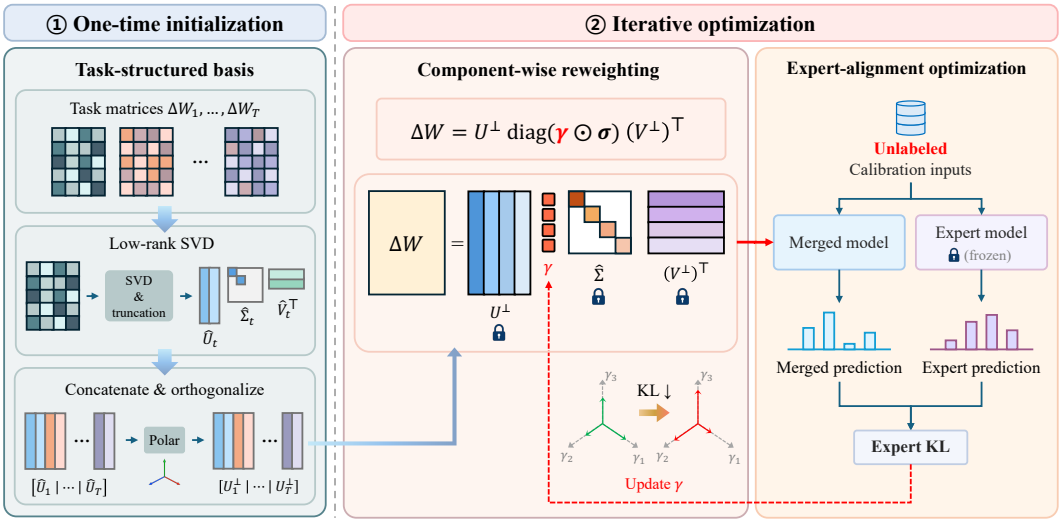


Figure 2: Overview of our method. Spectral components form a fixed, task-structured orthogonal basis; non-negative coefficients are learned by matching expert predictions on unlabeled calibration inputs.

task-wise or layer-wise coefficients, AdaRank learns binary masks and layer-wise coefficients, and SWIM-Merging adapts post-merge spectral coefficients. Expert Merging (Zhang et al., 2026) instead learns layer-wise or chunk-wise coefficients from unlabeled calibration data by aligning the merged model's hidden states and logits with those of the corresponding experts.

3 Preliminaries and Approach

3.1 Task-Structured Orthogonal Basis

We consider T task-specific models fine-tuned from a shared pre-trained model and merge them layer by layer. For an arbitrary weight matrix of size $d_{\text{out}} \times d_{\text{in}}$, we omit the layer index for clarity. Ilharco et al. (2023) define a task vector as the difference between fine-tuned and pre-trained model parameters. For a weight matrix in a given layer, Gargiulo et al. (2025) define the corresponding task matrix as $\Delta \boldsymbol{W}_t = \boldsymbol{W}_t - \boldsymbol{W}_0$, where $\boldsymbol{W}_t$ and $\boldsymbol{W}_0$ denote the fine-tuned and pre-trained weight matrices, respectively. They apply SVD to each task matrix $\Delta \boldsymbol{W}_t$ and retain its top-k singular components, where $k = \lfloor r/T \rfloor$ and $r = \min(d_{\text{out}}, d_{\text{in}})$. We denote the resulting factors by $\hat{\boldsymbol{U}}_t$, $\hat{\boldsymbol{\Sigma}}_t$, and $\hat{\boldsymbol{V}}_t$.

The truncated singular-vector matrices are concatenated across tasks as $\hat{\boldsymbol{U}} = [\hat{\boldsymbol{U}}_1, \ldots, \hat{\boldsymbol{U}}_T]$ and $\hat{\boldsymbol{V}} = [\hat{\boldsymbol{V}}_1, \ldots, \hat{\boldsymbol{V}}_T]$. The singular-value matrices $\hat{\boldsymbol{\Sigma}}_1, \ldots, \hat{\boldsymbol{\Sigma}}_T$ form the diagonal blocks of $\hat{\boldsymbol{\Sigma}}$. Because the retained subspaces from different tasks are generally non-orthogonal, TSV-M (Gargiulo et al., 2025) performs cross-task orthogonalization by taking the polar factor of $\hat{\boldsymbol{U}}$, denoted by $\boldsymbol{U}^{\perp}$. Applying the same procedure to $\hat{\boldsymbol{V}}$ yields $\boldsymbol{V}^{\perp}$. The merged task matrix is $\Delta \boldsymbol{W}_{\text{merge}} = \boldsymbol{U}^{\perp} \hat{\boldsymbol{\Sigma}} (\boldsymbol{V}^{\perp})^{\top}$. The merged weight at each layer is $\boldsymbol{W}_0 + \alpha \Delta \boldsymbol{W}_{\text{merge}}$, where α is a global scaling coefficient. Our method fixes $\alpha = 1$ and learns component-wise coefficients instead.

3.2 Expert-Aligned Component-wise Reweighting

Figure 2 summarizes our method: we fix a task-structured orthogonal basis and jointly learn non-negative coefficients for its spectral components by matching expert predictions on unlabeled calibration inputs.

Component-wise Coefficient Learning. We assign an independent coefficient γ_j to each orthogonal component. Let $m = Tk$ denote the total number of retained components for a weight matrix.

Omitting the layer index, the reweighted update for a single weight matrix is

$$\Delta \boldsymbol{W}(\boldsymbol{\gamma}) = \boldsymbol{U}^{\perp} \operatorname{diag}(\boldsymbol{\gamma} \odot \boldsymbol{\sigma})(\boldsymbol{V}^{\perp})^{\top} = \sum_{j=1}^{m} \gamma_j \sigma_j \mathbf{u}_j^{\perp} (\mathbf{v}_j^{\perp})^{\top}, \tag{1}$$

where $\boldsymbol{\sigma} = \operatorname{diag}(\hat{\boldsymbol{\Sigma}})$ collects the retained singular values.

The merged weight matrix is

$$\boldsymbol{W}_{\text{merged}} = \boldsymbol{W}_0 + \Delta \boldsymbol{W}(\boldsymbol{\gamma}). \tag{2}$$

Only the coefficients are learned; the orthogonal basis directions and retained singular values remain fixed.

Expert-aligned Optimization. We jointly optimize the coefficients across all merged weight matrices, collectively denoted by $\boldsymbol{\gamma}$. For each task t, we split an unlabeled calibration set $\mathcal{D}_t^{\text{cal}}$ into disjoint optimization and holdout subsets, $\mathcal{D}_t^{\text{opt}}$ and $\mathcal{D}_t^{\text{hold}}$. We optimize the coefficients on $\mathcal{D}_t^{\text{opt}}$ and use task-averaged Expert KL on $\mathcal{D}_t^{\text{hold}}$ for early stopping. The objective is the task-averaged forward KL divergence (Hinton et al., 2015):

$$\mathcal{L}(\boldsymbol{\gamma}) = \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_{x \sim \mathcal{D}_t^{\text{opt}}} \left[D_{\text{KL}} \left(p_t^{\text{expert}}(x) \parallel p_t^{\text{merged}}(x; \boldsymbol{\gamma}) \right) \right]. \tag{3}$$

Both predictive distributions are computed using temperature-scaled softmax with temperature τ. Frozen expert predictions provide targets for jointly optimizing the coefficients under a non-negativity constraint:

$$\boldsymbol{\gamma}^{\star} = \arg\min_{\boldsymbol{\gamma} \geq \mathbf{0}} \mathcal{L}(\boldsymbol{\gamma}). \tag{4}$$

Allowing signed real-valued coefficients yields similar performance, so we retain the non-negativity constraint to prevent sign reversal of task-derived components (Appendix B.3).

4 Why Component-wise Coefficients?

We investigate why reweighting is needed after cross-task orthogonalization, why its granularity matters, and whether component-wise adjustment reduces cross-task interference.

4.1 Residual Expert Discrepancy after Orthogonalization

Cross-task orthogonalization eliminates the pairwise overlap between the retained task subspaces, but leaves expert discrepancy that can be reduced by reweighting the fixed components. We quantify the overlap between the retained left singular subspaces of distinct tasks t and t' using $\|\hat{\boldsymbol{U}}_t^{\top} \hat{\boldsymbol{U}}_{t'}\|_F^2 / k$, following Hu et al. (2021). This quantity equals the mean squared cosine of the principal angles between the column spaces of $\hat{\boldsymbol{U}}_t$ and $\hat{\boldsymbol{U}}_{t'}$. We average it over task pairs and layers. Because $\boldsymbol{U}^{\perp}$ has orthonormal columns, its task-specific column blocks have zero pairwise overlap.

We next examine whether eliminating this overlap also aligns the merged model with the task experts. For each task t, let p_t^{expert} and p_t^{merged} denote the predictive distributions of the expert and the merged model. We measure expert discrepancy using Expert KL, defined as $D_{\text{KL}}(p_t^{\text{expert}} \parallel p_t^{\text{merged}})$ averaged over test inputs, and top-1 disagreement, defined as the percentage of test inputs for which the two models predict different classes. Both metrics are averaged over tasks.

Table 1: Expert discrepancy before and after cross-task orthogonalization and after subsequent reweighting on ViT-B/32 with 8 tasks.

	Orthogonalization		+ Reweighting	
	Before	After	Layer × Task	Component
Task-subspace overlap	Non-zero	0.0	0.0	0.0
Expert KL	0.96	0.43	0.30	0.19
Top-1 disagreement (%)	30.33	14.80	10.79	7.78

Table 1 shows that orthogonalization eliminates task-subspace overlap, yet leaves substantial expert discrepancy. With the orthogonal basis fixed, learning one shared coefficient per layer–task group[1]

[1]Figure 3 illustrates coefficient sharing at different granularities. We refer to components originating from the same task and layer as a layer–task group.

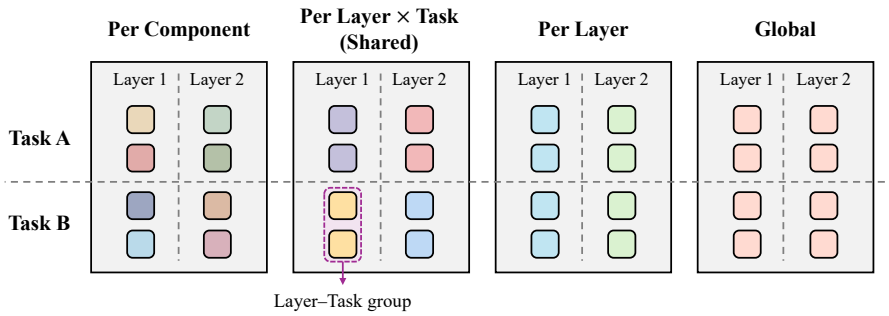


Figure 3: Coefficient sharing at four granularities. Components with the same color share one coefficient.

on a separate calibration set reduces test-set Expert KL from 0.43 to 0.30 and top-1 disagreement from 14.80% to 10.79%. Learning a separate coefficient for each component further reduces Expert KL and top-1 disagreement to 0.19 and 7.78%, respectively. Both parameterizations preserve zero task-subspace overlap.

These results show that **1)** cross-task orthogonalization alone leaves substantial expert discrepancy, **2)** Expert KL correlates with top-1 disagreement, and **3)** reweighting granularity matters. To investigate why finer-grained reweighting is more effective, we next examine how individual components within the same layer–task group affect expert discrepancy across tasks.

4.2 Heterogeneous Component Effects

We observe that components within the same layer–task group can have different cross-task effects, motivating adjustments beyond a shared group scale. A shared coefficient scales all components in a group equally and therefore preserves their relative magnitudes. To examine whether individual components favor different adjustments, we measure how increasing each component's coefficient changes Expert KL across tasks[2].

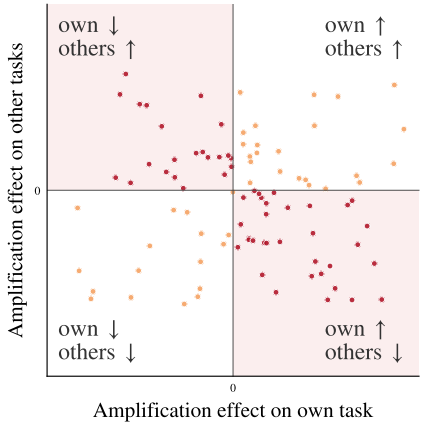


Figure 4: Source-task and aggregated other-task Expert KL effects for 96 components in one Cars layer–task group.

Let γ_j scale the j-th spectral component, with $\boldsymbol{\gamma} = \mathbf{1}$ representing the orthogonalized merge before reweighting. At this point, we compute the derivatives of Expert KL with respect to γ_j, holding all other coefficients fixed. We plot the negative derivatives so that positive values indicate that a small increase in γ_j reduces Expert KL. For the other tasks, we sum their derivatives; both source-task and other-task contributions are divided by T to match the task-averaged objective.

Figure 4 shows heterogeneous effects among components within a single layer–task group. Of the 96 components, 58% fall in the red regions, where the source-task and aggregate other-task derivatives have opposite signs. For these components, a small coefficient increase reduces source-task Expert KL but increases aggregate other-task Expert KL, or vice versa (all tasks in Appendix B.1). Thus, components from the same source task and layer need not affect expert discrepancy in the same way. A shared coefficient couples these distinct effects through a uniform change in scale. Within a group, amplifying some components reduces Expert KL on both the source and other tasks while amplifying others increases it on both, yet a shared coefficient must scale them in the same direction.

These observations suggest that a single shared scale may be too restrictive and that the relative weighting of components within a group may matter. In Section 4.3, we isolate this effect from changes in the overall group scale by fixing each group's optimized mean coefficient and testing whether redistributing weight across its components improves the merged model.

[2]Appendix B.4 shows that components retain substantial geometric correspondence with their index-assigned source tasks after polar orthogonalization, supporting the source-task interpretation in our analysis.

Table 2: Cross-task interference under two coefficient parameterizations on ViT-B/32 with 8 tasks. Source-Task Only keeps only the evaluated task's components; All-Task Merge keeps all. Values are task-averaged recovered fractions, computed per task as $(\mathrm{acc} - \mathrm{acc}_{\mathrm{zs}})/(\mathrm{acc}_{\mathrm{ft}} - \mathrm{acc}_{\mathrm{zs}})$, where $\mathrm{acc}_{\mathrm{zs}}$ and $\mathrm{acc}_{\mathrm{ft}}$ denote zero-shot and fine-tuned expert accuracy, respectively (0: zero-shot; 1: expert).

Coefficient Parameterization	Source-Task Only ↑	All-Task Merge ↑
Layer × Task (Shared) Scaling	0.948	0.838
Component Redistribution	0.945	0.902

4.3 Component Redistribution Reduces Cross-Task Interference

To isolate the effect of within-group allocation from group-level scaling, we first optimize one shared coefficient per layer–task group. We then redistribute the coefficients within each group while constraining their mean to remain at the optimized shared value. This controlled comparison tests whether adjusting relative component weights provides a benefit beyond optimized shared scaling.

Figure 5 illustrates this controlled comparison. Let $\mathcal{S}$ denote the shared-coefficient subspace in which coefficients are equal within each group, and let Π be the orthogonal projection onto $\mathcal{S}$. Starting from the optimized shared-coefficient solution $\boldsymbol{\gamma}_{\mathrm{LT}}$, we learn $\boldsymbol{\gamma} = \boldsymbol{\gamma}_{\mathrm{LT}} + \boldsymbol{\delta}$ subject to $\Pi\boldsymbol{\delta} = \mathbf{0}$. Equivalently, $\sum_{j \in \mathcal{B}_g} \delta_j = 0$ for every group $\mathcal{B}_g$, so the adjustment preserves each group's mean and lies in the orthogonal complement of $\mathcal{S}$ in coefficient space. This isolates adjustments unavailable to shared scaling while holding the learned group averages fixed. The mean-preserving constraint is used only for this analysis, not for our general component-wise optimization.

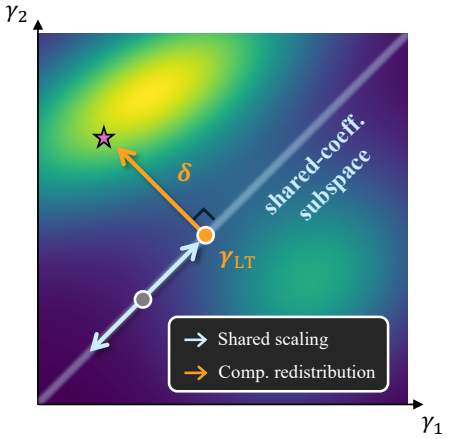


Figure 5: Schematic of coefficient adjustment for two components in the same layer–task group. Brighter background regions indicate lower Expert KL. Gray circle: initialization; orange circle: shared-coefficient solution; purple star: after redistribution.

Table 2 shows that redistribution improves All-Task Merge from 0.838 to 0.902, while Source-Task Only remains nearly unchanged (0.948 to 0.945). The gap narrows from 0.110 to 0.043 (14 and 20 tasks in Appendix B.2). These results show that within-group redistribution provides a benefit beyond optimized shared scaling: even after the group-level scale is fixed, the relative allocation of magnitude across components remains important for cross-task compatibility.

5 Experiments

5.1 Experimental Setup

Benchmarks and Models. Following the standard vision model merging protocol (Gargiulo et al., 2025; Li et al., 2026b), we evaluate our method using CLIP (Radford et al., 2021) with three backbones: ViT-B/32, ViT-B/16, and ViT-L/14, on benchmarks comprising 8, 14, and 20 tasks. The 8-task benchmark comprises Cars, DTD, EuroSAT, GTSRB, MNIST, RESISC45, SUN397, and SVHN. The 14-task benchmark adds CIFAR100, STL10, Flowers102, Oxford-IIIT Pet, PCAM, and FER2013. The 20-task benchmark further adds EMNIST, CIFAR10, Food101, FashionMNIST, RenderedSST2, and KMNIST. Appendix A.1 provides dataset references and checkpoint details.

Baselines. We group baselines according to the information available during merging. *Data-free* methods operate only on model parameters or task vectors: Task Arithmetic (Ilharco et al., 2023), TIES-Merging (Yadav et al., 2023), TSV-M (Gargiulo et al., 2025), Iso-CTS (Marczak et al., 2025), and TSV-M with SVC (Li et al., 2026c). *Calibration-based* methods, including ESM (Li et al., 2026b) and Expert Merging++ (Zhang et al., 2026), additionally use a small set of unlabeled calibration inputs. Since Expert Merging++ was originally evaluated on LLMs and MLLMs, we adapt it to the CLIP-based vision setting.

Table 3: Average accuracy (%) over tasks. In parentheses, normalized accuracy (%): each task's accuracy divided by that of its fine-tuned expert, averaged over tasks. Calib. indicates the use of a calibration set during merging. A dash denotes a reference model.

		ViT-B/32			ViT-B/16			ViT-L/14		
Method	**Calib.**	8 tasks	14 tasks	20 tasks	8 tasks	14 tasks	20 tasks	8 tasks	14 tasks	20 tasks
Zero-shot	—	47.6 (54.3)	56.8 (64.1)	57.5 (64.4)	54.3 (59.4)	60.7 (66.2)	61.6 (66.9)	64.5 (69.0)	68.1 (72.7)	67.6 (71.9)
Fine-tuned	—	90.5	89.5	90.4	92.6	91.6	92.4	94.1	93.3	94.0
Task Arithmetic	✗	68.4 (75.6)	64.4 (72.0)	60.8 (67.8)	73.6 (79.3)	69.6 (75.6)	65.3 (70.8)	82.7 (87.7)	78.6 (83.8)	73.2 (77.7)
TIES-Merging	✗	72.0 (79.6)	66.7 (74.5)	62.7 (69.7)	77.2 (83.2)	72.0 (78.2)	67.4 (73.1)	84.4 (89.7)	78.4 (83.8)	75.0 (79.6)
TSV-M	✗	82.4 (90.9)	78.5 (87.5)	75.8 (83.8)	86.6 (93.3)	83.2 (90.6)	79.9 (86.3)	90.9 (96.6)	87.9 (94.1)	86.7 (92.0)
Iso-CTS	✗	82.5 (91.2)	79.5 (88.6)	76.7 (84.8)	88.6 (95.6)	84.6 (92.0)	81.2 (87.8)	92.7 (98.5)	89.4 (95.6)	89.1 (94.6)
TSV-M w/ SVC	✗	83.1 (91.8)	78.9 (87.9)	76.1 (84.0)	87.3 (94.1)	83.7 (91.2)	80.7 (87.2)	91.3 (97.0)	88.6 (94.8)	87.6 (93.0)
Expert Merging++	✓	82.7 (91.3)	80.4 (89.6)	81.0 (89.4)	86.0 (92.7)	83.5 (91.0)	83.8 (90.6)	90.6 (96.3)	88.4 (94.7)	89.1 (94.7)
ESM	✓	85.2 (94.3)	82.4 (92.1)	80.5 (89.0)	89.1 (96.2)	86.4 (94.3)	84.1 (91.0)	92.5 (98.4)	89.9 (96.3)	89.6 (95.3)
Ours	✓	**88.2 (97.5)**	**86.2 (96.1)**	**86.6 (95.7)**	**90.6 (97.8)**	**88.6 (96.6)**	**89.2 (96.5)**	**93.3 (99.2)**	**92.0 (98.6)**	**92.6 (98.4)**

Implementation Details. Unless otherwise specified, each task uses a calibration set of 256 unlabeled inputs sampled from the validation set and split into optimization and holdout subsets. We initialize all component coefficients as $\gamma_j = 1$, and jointly optimize them using Adam (Kingma & Ba, 2014) with a learning rate of 1×10^{-2}, temperature $\tau = 4$, and a mini-batch size of 32 (Appendix A.2). Our method fixes $\alpha = 1$ throughout without searching over α. For TSV-M and ESM, we select α using labeled validation inputs, following the validation-based tuning procedures described in the original papers (Gargiulo et al., 2025; Li et al., 2026b). We apply the same search to Expert Merging++. All calibration-based experiments are repeated three times with independently sampled calibration sets, and we report the mean. Unless otherwise noted, experiments are conducted on a single NVIDIA B200 GPU.

5.2 Results and Analysis

Main Results. Table 3 reports absolute and normalized accuracy across three CLIP backbones and 8, 14, and 20 tasks. Our method achieves the highest accuracy in all nine settings, and its margin over the strongest baseline increases with task count for each backbone. This trend is consistent with increased cross-task interactions in larger merges: our objective jointly optimizes all component coefficients across tasks and layers against the expert predictions. It retains 95.7–99.2% of the corresponding fine-tuned expert accuracy across all settings.

Ablation of Design Choices. Figure 3 illustrates the four coefficient-sharing granularities. Table 4 ablates coefficient granularity, objective, and optimization strategy on ViT-B/32. Accuracy improves as coefficient sharing becomes finer, reaching its highest level with per-component coefficients. Expert KL outperforms entropy minimization and yields slightly higher accuracy than ground-truth cross-entropy, showing that expert predictions provide an effective label-free calibration signal. Per-task independent optimization degrades sharply as task count increases. Block-alternating optimization, which cycles through the 12 residual blocks and updates one block's coefficients, achieves accuracy comparable to joint optimization but requires more than $8\times$ as much optimization time. We therefore use joint optimization as the default to account for cross-task component effects while remaining computationally efficient.

Table 4: Ablation on ViT-B/32 (average absolute accuracy, %). Each block varies one design choice while fixing the others to our default setting: per-component coefficients, Expert KL, and joint optimization.

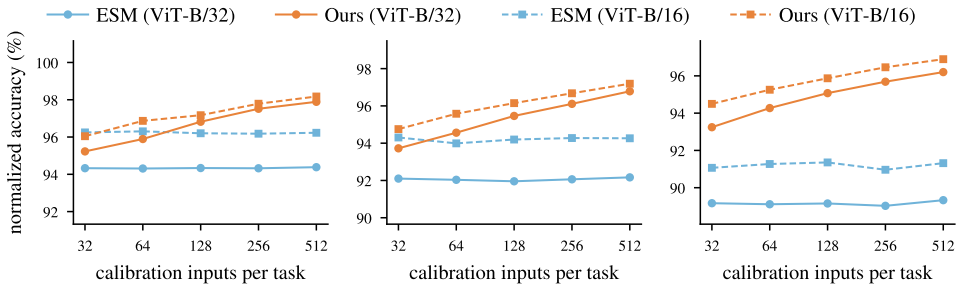
Method	8 tasks	14 tasks	20 tasks
Granularity			
Global	82.23	78.20	75.71
Per Layer	84.73	81.56	80.15
Per Layer × Task	85.91	83.68	83.73
Per Component (Ours)	**88.23**	**86.15**	**86.64**
Objective			
Entropy min.	77.81	72.67	66.87
GT-label CE	87.91	85.33	85.84
Expert KL (Ours)	**88.23**	**86.15**	**86.64**
Optimization			
Per-task independent	76.00	62.15	46.42
Block-alternating	**88.32**	86.12	86.59
Joint (Ours)	88.23	**86.15**	**86.64**

Table 5: Spearman correlations of σ_j with γ_j^* and $\gamma_j^*\sigma_j$, computed within and averaged across layer–task groups on ViT-B/32.

	8 tasks	14 tasks	20 tasks
Spearman(σ_j,γ_j^*)	−0.070	−0.094	−0.099
Spearman(σ_j,$\gamma_j^*\sigma_j$)	0.242	0.162	0.141

Table 6: Basis ablation on ViT-B/32 (average absolute accuracy, %). $\gamma = \mathbf{1}$ denotes coefficients fixed at initialization; 32, 128, and 256 denote results after coefficient learning with that many calibration inputs per task.

	8 tasks				14 tasks				20 tasks			
Basis	γ=1	32	128	256	γ=1	32	128	256	γ=1	32	128	256
Sum task vectors, then SVD	25.2	84.4	86.9	87.4	9.2	78.6	83.9	84.8	10.7	44.7	83.2	84.5
Per-task SVD + QR	70.7	81.2	85.1	86.0	71.2	79.6	82.8	83.8	70.8	77.6	81.6	82.3
Per-task SVD + Polar	82.4	**86.4**	**87.6**	**88.2**	78.3	**83.9**	**85.6**	**86.2**	75.4	**84.3**	**86.1**	**86.6**



(a) 8 tasks (b) 14 tasks (c) 20 tasks

Figure 6: Normalized accuracy as the number of calibration inputs per task varies. Each point is the mean of three runs with independently sampled calibration sets.

Relationship to the Singular-value Spectrum. Table 5 shows weak average within-group Spearman correlations between the original singular values σ_j and learned coefficients γ_j^*. The correlations between the original singular values σ_j and reweighted magnitudes $\gamma_j^*\sigma_j$ are also low, indicating that reweighting substantially changes the original spectral ordering.

Choice of Orthogonal Basis. Table 6 compares three orthogonal basis constructions with the same number of trainable component coefficients. *Sum task vectors, then SVD* applies SVD to the summed task update, yielding components without an explicit source-task assignment. *Per-task SVD + QR* concatenates the retained singular vectors of each task and orthonormalizes them by QR decomposition, with column signs aligned to the corresponding input vectors. *Per-task SVD + Polar* uses the same concatenation and extracts its orthogonal polar factor, the matrix with orthonormal columns closest to it in Frobenius norm, following TSV-M.

Coefficient learning improves performance under all three constructions. The polar basis achieves the highest accuracy across all evaluated task counts and calibration set sizes. With 256 calibration inputs per task, its margin over the strongest alternative increases with task count. It also shows the smallest accuracy drop when the calibration set is reduced from 256 to 32 inputs per task. These results support our choice of the polar basis.

Calibration Data Efficiency. Our method outperforms ESM in all evaluated settings except on ViT-B/16 with 8 tasks and 32 calibration inputs per task (Figure 6). Its performance improves with larger calibration sets, whereas ESM shows little change across calibration budgets.

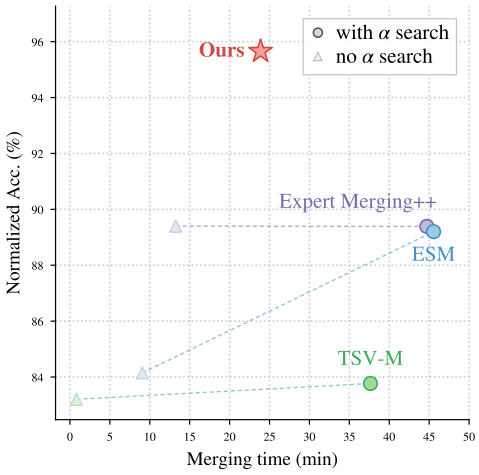


Figure 7: Merging cost versus accuracy on ViT-B/32 with 20 tasks, evaluated on a single NVIDIA RTX PRO 6000 GPU. The upper left is better.

Merging Cost. For baselines that tune a global merge coefficient α, the search requires labeled validation inputs and dominates the merging time. Our method uses no search over α and achieves higher accuracy than the baselines with α search (Figure 7). Omitting the search and fixing $\alpha = 1$ reduces ESM's merging cost but lowers its accuracy.

Table 7: MergeBench domain scores (%) for merging five experts fine-tuned from Llama-3.2-3B. Avg. is the arithmetic mean of the five domain scores.

Method	Instruction	Math	Coding	Multilingual	Safety	Avg.
Llama-3.2-3B	6.84	29.72	27.17	40.69	27.72	26.43
Individual experts	38.26	60.58	43.90	41.69	76.21	52.13
Task Arithmetic	24.95	50.49	41.29	**42.33**	45.56	40.92
TSV-M	18.30	54.74	40.44	42.11	48.07	40.73
Expert Merging	22.92	53.68	40.23	42.31	44.76	40.78
TSV-M w/ OrthoMerge-C	21.07	55.95	**41.69**	42.21	46.64	41.51
SAIM	32.72	57.32	38.18	41.04	47.66	43.38
Ours	**35.30**	**61.56**	38.96	41.70	**53.13**	**46.13**

Table 8: MergeBench domain scores (%) for merging five experts fine-tuned from Llama-3.2-3B-Instruct. Avg. is the arithmetic mean of the five domain scores.

Method	Instruction	Math	Coding	Multilingual	Safety	Avg.
Llama-3.2-3B-Instruct	50.65	64.97	46.85	41.68	94.63	59.76
Task Arithmetic	42.14	73.24	52.93	41.99	93.19	60.70
TSV-M	52.87	75.06	52.37	42.00	94.33	63.33
Expert Merging	44.92	73.92	**53.16**	**42.21**	92.81	61.40
TSV-M w/ OrthoMerge-C	52.68	74.83	52.31	41.41	**95.07**	63.26
SAIM	46.21	71.95	50.98	41.11	92.25	60.50
Ours	**55.45**	**76.12**	52.11	41.91	94.91	**64.10**

5.3 Extension to Language Models

Benchmark and Setup. We merge the five domain experts released by MergeBench (He et al., 2025) for each of two base models, Llama-3.2-3B and Llama-3.2-3B-Instruct (Grattafiori et al., 2024). Each domain score averages the primary metrics of its benchmarks, and the overall score averages the five domain scores. We apply the same component-wise reweighting to the attention and MLP matrices with $\alpha = 1$, matching expert next-token distributions at response positions on up to 256 calibration inputs per domain; reference responses serve only as conditioning context, not as targets. We report a single run and compare against Task Arithmetic, TSV-M, Expert Merging (Zhang et al., 2026), TSV-M with OrthoMerge-C (Yang et al., 2026b), and SAIM (Yang et al., 2026a) (Appendix A.3).

Results. Tables 7 and 8 report the MergeBench results. Among the evaluated merging methods, ours achieves the highest average score under both initializations. For Llama-3.2-3B, it reaches 46.13 compared with 43.38 for the strongest baseline, SAIM, with the largest gains in instruction following, mathematics, and safety; for Llama-3.2-3B-Instruct, it reaches 64.10 compared with 63.33 for TSV-M. It also achieves the highest instruction-following and mathematics scores under both, while the best-performing methods for coding and multilingual understanding vary across settings.

6 Conclusion

Cross-task orthogonalization leaves expert discrepancy, and spectral components within the same layer–task group have heterogeneous cross-task effects. We address this with spectral component-wise reweighting for expert alignment in a fixed orthogonal basis. Mean-preserving redistribution supports the benefit of adjusting relative component weights, and our method achieves the highest average performance among the evaluated merging methods across nine vision and two language-model settings.

Limitations and Future Work. Our approach requires task-specific calibration inputs and access to expert predictions. Future work should reduce these requirements and investigate robustness to calibration–deployment distribution shifts. Another direction is to learn orthogonality-preserving rotations of the spectral directions jointly with their coefficients.

AI USE STATEMENT

Generative AI tools were used to assist with language refinement, LaTeX formatting, retrieval of related work, and feedback on the manuscript's logical flow, organization, and consistency of terminology and notation. All AI-assisted suggestions were reviewed by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Section 3 describes the basis construction and coefficient-learning objective. Section 5.1 and Appendix A provide the experimental setup, checkpoint information, and implementation details, including calibration splits, optimization settings, and early stopping. Additional experimental details and analyses are provided in Appendix B.

REFERENCES

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017. doi: 10.1109/JPROC.2017.2675998.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *CoRR*, abs/1812.01718, 2018. URL http://arxiv.org/abs/1812.01718.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Guodong Du, Junlin Lee, Jing Li, Runhua Jiang, Yifei Guo, Shuyang Yu, Hanting Liu, Sim K Goh, Ho-Kin Tang, Daojing He, et al. Parameter competition balancing for model merging. *Advances in Neural Information Processing Systems*, 37:84746–84776, 2024.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodola. Task singular vectors: Reducing task interference in model merging. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18695–18705. IEEE, 2025.

Ian J. Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron C. Courville, Mehdi Mirza, Benjamin Hamner, William Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, Yingbo Zhou, Chetan Ramaiah, Fangxiang Feng, Ruifan Li, Xiaojie Wang, Dimitris Athanasakis, John Shawe-Taylor, Maxim Milakov, John Park, Radu Tudor Ionescu, Marius Popescu, Cristian Grozea, James Bergstra, Jingjing Xie, Lukasz Romaszko, Bing Xu, Chuang Zhang, and Yoshua Bengio. Challenges in representation learning: A report on three machine learning contests. *Neural Networks*, 2015.

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.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized llms. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-3374. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/91f7f71cb04699f387dc863da42a1fe3-Paper-Datasets_and_Benchmarks_Track.pdf.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *JS-TARS*, 2019.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic, 2023. URL https://arxiv.org/abs/2212.04089.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chanhyuk Lee, Jiho Choi, Chanryeol Lee, Donggyun Kim, and Seunghoon Hong. Adarank: Adaptive rank pruning for enhanced model merging. In *International Conference on Learning Representations*, volume 2026, pp. 6484–6510, 2026.

Boqi Li, Jingyuan Xu, Weiwei Liu, and Xiuwen Gong. Swim-merging: Reweighting singular values of weight matrices for efficient test-time model merging. *Neurocomputing*, 704:134823, 2026a. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2026.134823. URL https://www.sciencedirect.com/science/article/pii/S0925231226022216.

Longhua Li, Lei Qi, Qi Tian, and Xin Geng. Model merging in the essential subspace. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 31145–31154, June 2026b.

Yayuan Li, Ze Peng, Jian Zhang, Jintao Guo, Yue Duan, and Yinghuan Shi. When shared knowledge hurts: Spectral over-accumulation in model merging, 2026c. URL https://arxiv.org/abs/2602.05536.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D. Bagdanov, and Joost Van De Weijer. No task left behind: Isotropic model merging with common and task-specific subspaces. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 43177–43199. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/marczak25a.html.

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

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170/.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, pp. 1453–1460. IEEE, 2011.

Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 210–218. Springer, 2018.

Ke Wang, Nikolaos Dimitriadis, Guillermo Ortiz-Jimenez, François Fleuret, and Pascal Frossard. Localizing task information for improved model merging and compression. *arXiv preprint arXiv:2405.07813*, 2024.

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

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. URL https://arxiv.org/abs/1708.07747.

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

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*, 36:7093–7115, 2023.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations*, volume 2024, pp. 22743–22763, 2024.

Qun Yang, Enneng Yang, Wei Chen, Li Shen, and Long Lan. Merge to remember: Sharpness-aware isotropic merging for continual learning. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=aWfk16EvfM.

Sihan Yang, Kexuan Shi, and Weiyang Liu. Orthogonal model merging. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=pCaeMsqpCa.

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 Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57755–57775. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/yu24p.html.

Dengming Zhang, Xiaowen Ma, Zhen-Liang Ni, Zhenkai Wu, Han Shu, Xin Jiang, and Xinghao Chen. Expert merging: Model merging with unsupervised expert alignment and importance-guided layer chunking. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Awf3ebMpKw.

A Experimental Setup and Implementation Details

A.1 Vision Datasets and Checkpoints

Datasets. The original references for the vision datasets are listed below. The 14- and 20-task benchmarks cumulatively add datasets to the preceding benchmark.

- **8 tasks:** Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2011), MNIST (LeCun et al., 1998), RESISC45 (Cheng et al., 2017), SUN397 (Xiao et al., 2010), and SVHN (Netzer et al., 2011).
- **14 tasks:** The 8-task benchmark plus CIFAR100 (Krizhevsky, 2009), STL10 (Coates et al., 2011), Flowers102 (Nilsback & Zisserman, 2008), Oxford-IIIT Pet (Parkhi et al., 2012), PCAM (Veeling et al., 2018), and FER2013 (Goodfellow et al., 2015).
- **20 tasks:** The 14-task benchmark plus EMNIST (Cohen et al., 2017), CIFAR10 (Krizhevsky, 2009), Food101 (Bossard et al., 2014), FashionMNIST (Xiao et al., 2017), RenderedSST2 (Socher et al., 2013), and KMNIST (Clanuwat et al., 2018).

Checkpoints. All methods use the same fine-tuned checkpoints released by Wang et al. (2024). Normalized accuracy is computed by dividing, for each task, the merged model's test accuracy by the test accuracy of the corresponding fine-tuned expert checkpoint, and then averaging these ratios across tasks. Published absolute accuracies are not directly comparable when different fine-tuned checkpoints are used. For example, the average fine-tuned accuracy on ViT-B/32 with 8 tasks is 90.5% in our setting and in SVC, whereas Iso-CTS and ESM use checkpoints with an average accuracy of 92.8%. We therefore rerun all baselines using our checkpoints rather than quoting their published results, ensuring that all methods in Table 3 are evaluated from the same fine-tuned experts. For cross-paper comparison, we report normalized accuracy to account for differences in fine-tuned expert performance. For DTD on ViT-B/32, we re-evaluate the released checkpoint and use its measured accuracy of 78.99%, as the value in `nonlinear_ft_accuracies.json` is inconsistent with the checkpoint.

A.2 Vision Implementation details

Calibration Inputs and Early Stopping. For each task, we sample calibration inputs from the validation set and partition them into disjoint optimization and holdout subsets. Given a budget of N inputs per task, the holdout size m and the optimization size n are defined as

$$m = \min\left(\frac{N}{4}, 32\right), \qquad n = N - m, \tag{5}$$

such that one quarter of the calibration budget is reserved for early stopping when $N \leq 128$, while the holdout size is capped at 32 for larger budgets. Table 9 lists the resulting splits. Unless otherwise stated, we use $N = 256$, yielding $n = 224$ optimization inputs and $m = 32$ holdout inputs per task.

Table 9: Calibration split per task for each budget N.

N	n	m
32	24	8
64	48	16
128	96	32
256	224	32
512	480	32

We evaluate the holdout Expert KL every 40 optimization steps, and treat any strict decrease as an improvement. Optimization stops after 40 consecutive evaluations without improvement, corresponding to 1,600 optimization steps, with a maximum of 8,000 steps. We retain the coefficients from the step with the lowest holdout Expert KL rather than those from the final optimization step.

Temperature. Expert KL is computed at temperature τ, applied to both the expert and the merged model. As shown in Table 10, accuracy peaks at $\tau = 4$ for every task count and the margin over $\tau = 1$ grows with the number of tasks, so we use $\tau = 4$ throughout the vision experiments.

Full-batch vs. Mini-batch. Mini-batch updates use a fixed number of inputs per task from the optimization subsets, whereas full-batch updates use all inputs in those subsets. With 32 inputs per task, mini-batch optimization matches the accuracy of full-batch optimization at about one-third of the computation time, and batch sizes of 32 and 64 perform similarly (Table 11). We therefore use a batch size of 32 per task.

Table 10: Temperature ablation on ViT-B/32 with 512 calibration inputs per task. Absolute accuracy (%) is reported with normalized accuracy (%) in parentheses.

τ	8 tasks	14 tasks	20 tasks
1	87.84 (97.08)	85.31 (95.19)	82.78 (91.75)
2	88.11 (97.40)	86.24 (96.26)	84.10 (93.17)
4	**88.56 (97.89)**	**86.73 (96.78)**	**87.10 (96.20)**
8	88.14 (97.37)	86.05 (95.96)	84.64 (93.68)

Table 11: Effect of optimization batch size on ViT-B/32 with 8 tasks. Absolute accuracy (normalized accuracy) is reported in percent. All results, including time, are measured on a single NVIDIA RTX PRO 6000 GPU.

Sampling	Acc. (Norm.)	Time (min)
Mini-batch, 32 per task	**88.29 (97.60)**	**20.7**
Mini-batch, 64 per task	**88.29 (97.60)**	26.0
Full batch	88.09 (97.29)	61.1

A.3 Language Model Setup

Objective and Optimization. Calibration inputs come from the MergeBench validation split, and 12.5% of them are held out for model selection. We remove the 64 padded vocabulary entries of the experts and renormalize their distributions over the base vocabulary. The training loss sums the token-level KL over the response positions of each sequence, averages it over sequences, and sums it over domains; the holdout Expert KL used for model selection averages over tokens. We initialize all coefficients as $\gamma_j = 1$ and optimize them jointly with Adam (learning rate 10^{-2}), using temperature $\tau = 1$. We train for at most 2,000 steps and keep the coefficients with the lowest holdout Expert KL, evaluated every 20 steps. We use the MergeBench benchmarks and follow the metric choices of OrthoMerge (Yang et al., 2026b).

Baselines. Task Arithmetic uses $\lambda = 0.4$ from the MergeBench release, and TSV-M uses $\alpha = 1$. OrthoMerge-C uses the authors' released code, and SAIM uses its released merging operator ported to MergeBench, both with default settings. We reimplement Expert Merging, whose public code supports only InternVL, with one coefficient per parameter tensor for each expert, using the same calibration data and KL term as ours together with its intermediate hidden-state loss. Its chunked extension did not change the merged weights, so we report it as Expert Merging.

B Additional Experiments and Analyses

B.1 Heterogeneous Component Effects

We extend the analysis in Section 4.2 to all eight source tasks on ViT-B/32. Let $\mathcal{L}_t$ denote Expert KL averaged over the inputs of task t, and let $s(j)$ denote the source task of spectral component j. At the orthogonalized merge $\boldsymbol{\gamma} = \mathbf{1}$, we define

$$b_j^{\text{own}} = -\frac{1}{T}\left.\frac{\partial \mathcal{L}_{s(j)}}{\partial \gamma_j}\right|_{\boldsymbol{\gamma}=\mathbf{1}}, \qquad b_j^{\text{other}} = -\frac{1}{T}\left.\sum_{t \neq s(j)} \frac{\partial \mathcal{L}_t}{\partial \gamma_j}\right|_{\boldsymbol{\gamma}=\mathbf{1}}. \tag{6}$$

Positive values indicate that a small increase in γ_j reduces the corresponding Expert KL. Both contributions are scaled by $1/T$, so their sum equals the negative derivative of the task-averaged objective.

Figure 8 plots these contributions for components from each source task. The shaded quadrants contain components whose amplification reduces source-task Expert KL while increasing aggregated other-task Expert KL, or vice versa. Such opposing effects occur in all eight panels, extending the observation beyond Cars. The distributions also vary across source tasks. These results support examining component-specific adjustments, although the local effects at $\boldsymbol{\gamma} = \mathbf{1}$ do not determine the coefficients learned through joint optimization.

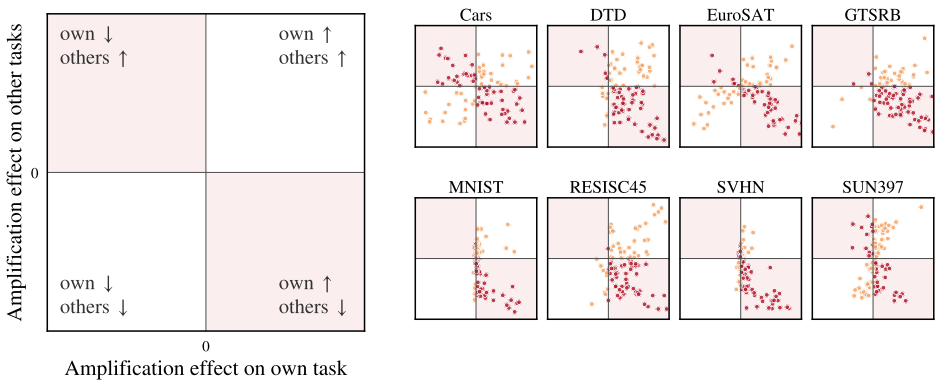


Figure 8: Effects of increasing spectral component coefficients on source-task and aggregated other-task Expert KL at $\boldsymbol{\gamma} = \mathbf{1}$. Each panel shows one source task; each point represents one component. Shaded regions indicate opposing effects.

Table 12: Task-averaged recovered fractions on ViT-B/32. Each cell reports Source-Task Only / All-Task Merge.

Coefficient Parameterization	8 tasks	14 tasks	20 tasks
Layer × Task Scaling	0.948 / 0.838	0.966 / 0.936	0.933 / 0.926
Component Redistribution	0.945 / 0.902	0.968 / 0.956	0.940 / 0.951

B.2 Within-Group Component Redistribution

We extend the analysis in Section 4.3 to 14 and 20 tasks and provide per-task results for the 8-task setting, using the same evaluation protocol. The per-task and task-averaged recovered fractions are

$$R_t = \frac{\mathrm{acc}_t - \mathrm{acc}_{t,\mathrm{zs}}}{\mathrm{acc}_{t,\mathrm{ft}} - \mathrm{acc}_{t,\mathrm{zs}}}, \qquad \bar{R} = \frac{1}{T}\sum_{t=1}^{T} R_t. \tag{7}$$

Table 12 reports results across task counts. Mean-preserving component redistribution improves All-Task Merge performance over layer–task sharing in all three settings.

Table 13 reports the 8-task results. Redistribution improves All-Task Merge performance on every task, with the largest gain on SUN397. Source-Task Only performance changes in both directions across tasks, while its average remains nearly unchanged.

B.3 Coefficient Constraints

We compare four coefficient domains while holding the remaining experimental settings fixed: $\{0, 1\}$ permits binary selection; $[0, 1]$ permits continuous suppression; $[0, \infty)$ permits both suppression and amplification; and $\mathbb{R}$ additionally permits sign reversal.

Table 14 shows that binary selection and continuous suppression differ by at most 0.16 percentage points in normalized accuracy. Allowing amplification improves normalized accuracy by 2.4–2.5 percentage points over continuous suppression across the three task counts. This supports learning coefficients that both suppress and amplify components. Allowing sign reversal yields similar accuracy to non-negative reweighting, with differences of at most 0.08 percentage points. We therefore use $[0, \infty)$, which supports component removal, suppression, and amplification while preventing sign reversal of the fixed spectral components.

Table 14: Coefficient constraint ablation on ViT-B/32. Values report normalized accuracy (%); all other experimental settings are held fixed.

	8 tasks	14 tasks	20 tasks
$\{0, 1\}$ (selection)	95.27	93.51	92.97
$[0, 1]$ (suppression)	95.14	93.61	93.13
$\mathbb{R}$	97.49	**96.15**	95.61
$[0, \infty)$	**97.52**	96.07	**95.65**

Table 13: Per-task recovered fractions on ViT-B/32 with 8 tasks. Each cell reports Source-Task Only / All-Task Merge.

	Cars	DTD	EuroSAT	GTSRB	MNIST	RESISC45	SVHN	SUN397
Layer × Task	0.928 / 0.758	0.972 / 0.895	0.925 / 0.890	0.973 / 0.969	0.969 / 0.978	0.969 / 0.904	0.908 / 0.892	0.941 / 0.416
Redistribution	0.906 / 0.830	1.008 / 0.946	0.907 / 0.908	0.973 / 0.979	0.962 / 0.985	0.955 / 0.939	0.899 / 0.944	0.952 / 0.683

Table 15: Geometric task provenance after polar orthogonalization on ViT-B/32. Cosine similarities and source-subspace energies are averaged over retained components across all reweighted weight matrices. Nearest-source agreement is computed separately for U and V; both yield 100% in every setting.

Metric	**8 tasks**	**14 tasks**	**20 tasks**
Mean same-index cosine (U)	0.86	0.84	0.84
Mean same-index cosine (V)	0.80	0.77	0.77
Mean source-subspace energy (U)	0.75	0.72	0.71
Mean source-subspace energy (V)	0.65	0.61	0.60
Nearest-source agreement (U and V)	100%	100%	100%

B.4 Geometric Task Provenance

Polar orthogonalization preserves the column indexing of the concatenated task-wise singular vectors, but can mix their directions. We therefore assign each component a source task by its original concatenation index and empirically assess its geometric correspondence to that source.

Geometric Correspondence. Let U_t contain the retained left singular vectors of task t, and let u_j and p_j^U denote the corresponding columns before and after polar orthogonalization. For n retained components in a layer, we define same-index cosine similarity, source-subspace energy, and nearest-source agreement as

$$\begin{aligned} c_j^U &= |\langle p_j^U, u_j \rangle|, \\ e_j^U &= \|U_{s(j)}^\top p_j^U\|_2^2, \\ \hat{s}^U(j) &= \arg\max_t \|U_t^\top p_j^U\|_2^2, \\ \text{Agreement}^U &= \frac{1}{n}\sum_{j=1}^{n} \mathbf{1}\left[\hat{s}^U(j) = s(j)\right], \end{aligned} \tag{8}$$

where $s(j)$ is the index-assigned source task. The right-factor metrics are defined analogously using V.

Table 15 shows substantial directional alignment, with approximately 71–75% of the left-factor energy retained in the original source-task subspace. Although alignment weakens slightly as more tasks are merged, the nearest source matches the assigned source for every evaluated component in both factors.