Appendix for PCB-Merging

A Novelty and Contribution

Our research aims to unlock the full potential of task vector-based approaches by adjusting coefficients at the parameter level through a balancing mechanism that addresses parameter competition across different tasks. We re-examine existing model merging methods and highlight the critical role of parameter competition awareness. To clearly demonstrate the innovation of our method, we conduct a comparative analysis with existing state-of-the-art baseline methods.

Comparison with TIES-Merging Both the TIES-Merging [86] and our approach address parameter competition or interference through self-awareness and cross-awareness. However, there are several key differences:

1. When performing *Drop* / *Trim* to reduce redundancy, we consider both intra-competition and inter-competition, whereas TIES-Merging primarily considers parameter magnitude.
2. In terms of cross-awareness, TIES-Merging only considers the direction of parameters across different tasks, neglecting parameter weights. Our method more accurately measures the similarity of task vectors to assess conflict levels. We conducted ablation experiments to demonstrate the effectiveness of inter-balancing, as shown in App. B.1 and Tab. 6.
3. Our approach modulates the coefficient of each parameter, while TIES-Merging uses a uniform scale for all tasks and parameters. Ablation experiments in the Analysis section validate the superiority of our method, as shown in Section 6.1 and Tab. 5.

Comparison with AdaMerging Although AdaMerging [87] has achieved significant performance improvements in image classification, it has several drawbacks:

1. This method requires unsupervised test samples, which is often impractical.
2. The use of Shannon entropy to train the adaptive weights limits the method to classification tasks.
3. AdaMerging requires unsupervised training with the availability of (unlabeled) test samples, which is a different setup than generalizing to an entirely unseen test set.

In contrast, our proposed PCB-Merging retains the efficiency and lightwight nature as most previous merging methods. Additionally, we conducted experiments on image classification tasks to compare the two methods, as shown in App. C.2 and Tab. 7.

Comparison with Fisher Merging and RegMean The same as Fisher Merging [43] and RegMean [27], our PCB-Merging method also introduces additional matrices to adjust parameter coefficients, but there are two key differences:

1. Fisher Merging and RegMean consider only self-awareness or cross-awareness, respectively. In contrast, our method accounts for various scenarios of parameter competition.
2. Both Fisher Merging and RegMean require additional gradient-based computations to obtain the Fisher Information Matrix or Inner Product Matrix, which demand more GPU resources. Our method, however, is based on task vectors, making it easier and lightwight to implement.

Comparison with DARE Both DARE [90] and PCB-Merging drop and rescale task vectors for model merging, but there are significant differences:

1. DARE randomly drops parameters according to a drop rate p, while we consider parameter competition.
2. DARE rescales the remaining parameters by a uniform factor of $1/(1-p)$, whereas we compute a specific coefficient for each task and each parameter.
3. DARE is mainly used in LLM model merging to maintain the original fine-tuned performance. In contrast, we find that dropping parameters can further enhance performance beyond the fine-tuned model with a suitable scale and intra-balancing.

Comparison with Lorahub Lorahub [23] aims to establish a strategic framework for composing LoRA modules trained on diverse tasks to achieve adaptable performance on new tasks. This framework utilizes an evolution algorithm (CMA-ES [19]) to search for the coefficients of each LoRA module, as introduced in Section 3.3. However, this search-based approach is time-consuming and can only be applied at the task level, leading to limited performance. Moreover, LoRA lacks self-awareness and considers only competition between different tasks.

Comparison with Task Arithmetic and PEM Compositon Both Task Arithmetic [26] and PEM Composition [92] methods primarily focus on exploring potential applications of task vectors, including distribution generalization, unlearning, and domain transfer. However, they do not address parameter competition or balance the coefficients of different tasks or parameters, which limits their performance.

Algorithm 1 PCB-Merging Procedure.

Input: Fine-tuned models $\{\theta_i\}_{i=1}^n$, Initialization θ_{pre}, mask ratio r and coefficient λ.
Output: Merged Model θ_m

▷ Create task vectors.
$\{\tau_i\}_{i=1}^n = \{\theta_i\}_{i=1}^n - \theta_{\text{pre}}$
for i ***in*** $1, ..., n$ **do**
  ▷ Step 1: Intra-Balancing.
  $\beta_{intra,i} = \text{Softmax}(N * \text{Norm}(\tau_i \odot \tau_i))$
  ▷ Step 2: Inter-Balancing.
  $\beta_{inter,i} = \sum_{j=1}^n \text{Softmax}(\tau_i \odot \tau_j)$
  ▷ Step 3: Drop low-scoring parameters.
  $\beta_i = \beta_{intra,i} \odot \beta_{inter,i}$
  $m_i = \beta_i \geq \text{sorted}(\beta_i)[(1-r) \times D]$
  $\hat{\beta}_i = m_i \odot \beta_i$
end
▷ Step 4: Rescale task vectors.
$\tau_m = \sum_{i=1}^n (\hat{\beta}_i \odot \tau_i) / \sum_{i=1}^n \hat{\beta}_i$
▷ Obtain merged checkpoint
$\theta_m \leftarrow \theta_{\text{init}} + \lambda * \tau_m$
return θ_m

B Additional Analysis

B.1 Additional Ablation Studies

We present additional ablation experiments on PCB-MERGING, as shown in Tab. 6. In addition to the four main steps discussed in Section 6.1 (Intra-Balancing, Inter-Balancing, Drop, and Rescale), we also tested other influencing factors:

1. Activation functions: We replaced the softmax activation function with common alternatives like sigmoid, ReLU, and tanh. The results show minimal performance loss with different activation functions, except for ReLU in intra-balancing. This is because these activation functions can represent complex nonlinear relationships to balance the values of parameters.
2. Without regulator N: We removed the regulator N in intra-balancing, which controls intra-competition according to the number of models being merged.
3. Inter-balancing with only sign: We computed inter-balancing using only the sign $(-1, 1)$ instead of the actual values, where the sign represents a direction in the D-dimensional parameter space relative to initialization. This experiment aims to compare with TIES-Merging, which addresses sign conflicts.
4. Element-wise multiplication vs. Addition: We combined intra-balancing and inter-balancing using addition instead of multiplication. This resulted in a performance loss of 4.1% and 3.9% on the ViT-B/32 and T5-base models, respectively.

In summary, these ablation experiments demonstrate the functionality and impact of each component in our method.

Table 6: More extensive ablation studies on PCB-MERGING

Ablation (→)	activation in intra-balancing			activation in inter-balancing			without	inter-balancing	replace multiplication	PCB
Model (↓)	sigmoid	relu	tanh	sigmoid	relu	tanh	regulator N	with only sign	by adding	Merging
ViT-B/32	76.1	74.9	76.1	76.2	76.1	76.4	74.7	75.7	72.2	**76.3**
T5-base	75.3	72.8	75.2	75.3	75.2	75.4	74.1	74.5	71.5	**75.4**

B.2 Additional Hyper-parameters Analysis

In this section, we present additional experimental results regarding hyper-parameters, observing similar phenomena and conclusions as those in Section 6.2. We explored the effects of λ and r on

the performance of merging multiple NLP tasks, as discussed in Section 5.1. First, we show the performance of various models for different values of λ, keeping $r = 0.2$. Our method is compared to the state-of-the-art baseline, TIES-Merging. As shown in Fig. 7, our approach achieves a higher performance ceiling within the optimal range of 0.8 to 1.6. As λ increases, the performance initially decreases and then levels off.

Furthermore, we provide a performance analysis for different values of r with T5-large. We conducted a grid search for λ to find its optimal performance for each ratio. Significantly, for $r < 0.4$, our method consistently shows substantial improvements. This highlights the importance of the information filtered by our parameter competition balancing approach in the merging process.

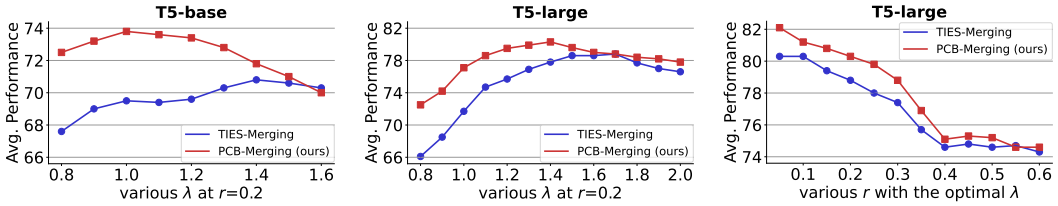

Figure 7: Performance with various hyperparameters λ and r.

C Additional Results

C.1 Merging Different Number of Tasks

We evaluated the performance of the merged model on in-domain tasks and analyzed how it varies with the number of tasks being merged. In Fig. 8, we normalized each task's accuracy to its fine-tuned model's performance and reported the average normalized accuracy for in-domain tasks with T5-base model. We compared our method against the strongest baseline, TIES-Merging [86], and simple averaging [83]. Each data point represents the merging of a subset of tasks, with the solid line indicating the average performance across multiple subsets. We observed that as the number of merged tasks increases, the performance of all methods declines, suggesting that more tasks lead to increased parameter competition. Additionally, TIES-Merging's performance drops faster than PCB-Merging, indicating that our PCB-Merging method is more effective in balancing parameter competition.

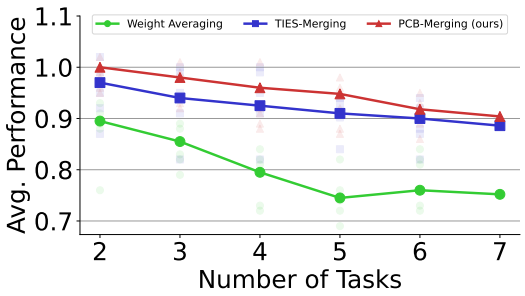

Figure 8: Average normalized performance when merging a different number of tasks.

C.2 Compare with Adamerging

We conducted cross-task merging experiments on image classification tasks to compare our method with AdaMerging [87]. AdaMerging employs unsupervised training to learn merging coefficients for each task vector in Task Arithmetic using unlabeled test datasets. Additionally, Layer-wise AdaMerging learns coefficients for each layer of each task vector.

Table 7: Compare the performance of different merging methods after applying unsupervised training with AdaMerging.

Model	Coefficient	AdaMerge	Ada + TIES	Ada + PCB
ViT-B/32	Task-wise	71.8	74.9	**77.1**
	Layer-wise	80.1	81.1	**81.7**
ViT-L/14	Task-wise	85.6	86.8	**88.2**
	Layer-wise	90.8	91.0	**91.3**

AdaMerging can be further improved by applying strategies from TIES-Merging to modify task vectors or using PCB-Matrix to adjust the task vectors. As shown in Tab. 7, our method enhances AdaMerging, resulting in performance improvements of 2.2% and 1.4% on the ViT-B/32 and ViT-L/14 models, respectively.

C.3 Compare with TIES-Merging using Evolutionary Strategy

To validate the effectiveness of the evolutionary strategy (ES) proposed in Section 3.3, we applied ES to intelligently search for coefficients of different tasks in other baseline methods. The results are shown in Tab. 8. Notably, after applying ES, TIES-Merging showed significant improvement. We also compared TIES-Merging with ES against our approach with ES. The results demonstrate the effectiveness of PCB-MERGING, particularly with a 2.2% performance gain on the T5-large model.

Table 8: Comparing the performance of different methods with evolutionary strategies (ES) after cross-task merging.

Task (→)	7 **NLP** Tasks		11 **PEFT** Tasks	3 **LLM** Tasks	8 **Vision** Tasks	
Method (↓)	T5-Base	T5-Large	$(IA)^3$	LLaMa2	ViT-B/32	ViT-L/14
Ties-Merging	73.6	80.3	66.8	34.2	73.6	86.0
PCB-MERGING (ours)	**75.4** (+1.8)	**82.1** (+1.8)	**68.1** (+1.3)	**35.1** (+0.9)	**76.4** (+2.8)	**87.5** (+1.5)
Ties-Merging + ES	74.8	81.0	67.6	34.3	74.9	86.8
PCB-MERGING + ES (ours)	**76.7** (+1.9)	**83.2** (+2.2)	**68.8** (+1.2)	**35.3** (+1.0)	**77.0** (+2.1)	**88.1** (+1.6)

C.4 Comprehensive Task-Level Results

We provide the task level for all the cross-task merging experiments in the main Tab. 2. Tab. 9, 10, 11, 12, and 13 provide the task level results T5-Base, T5-Large [56], IA3 [39], ViT-B/32, and ViT-L/14 [12] respectively. The task level results of the out-of-domain experiments for T5-Base and T5-Large can be found in Tab. 14.

Table 9: Test set performance when merging T5-base models on seven NLP tasks. Please refer to Section 5.1 for experimental details.

Task(→) / Method(↓)	Validation	Average	Test Set Performance						
			paws	qasc	quartz	story_cloze	wiki_qa	winogrande	wsc
Zeroshot	-	53.5	49.9	35.8	53.3	48.1	76.2	50	61.1
Fine-tuned	-	83.1	94.6	98.4	81.1	84.9	95.8	64.5	62.5
Multitask	-	83.6	94	97.9	82.5	86.7	95	64.1	65.3
Averaging[ICML22] [83]	✗	65.3	67.4	83.4	60.8	50.3	93.2	51.7	50.0
Task Arithmetic[ICLR23] [26]	✗	53.5	50.6	22.4	55.0	63.6	79.2	53.9	50.0
Ties-Merging[NeurIPS23] [86]	✗	69.5	76.1	79.5	68.5	65.6	86.3	56.2	54.2
PCB-MERGING (ours)	✗	73.8	77.1	91.5	**68.5**	75.8	88.2	**61.1**	54.2
Fisher Merging[NeurIPS22] [43]	✓	68.3	66.7	85.6	63.5	57.1	90.1	54.2	60.8
RegMean[ICLR23] [27]	✓	72.7	77.2	**93.8**	63.6	64.6	90.4	58.4	60.7
Task Arithmetic[ICLR23] [26]	✓	73.0	69.6	91.5	67.3	76.1	91.3	58.3	56.9
Ties-Merging[NeurIPS23] [86]	✓	73.6	**82.2**	84.8	66.1	73.5	87.0	60.2	61.1
PCB-MERGING (ours)	✓	**75.4**	79.0	93.2	65.8	**76.1**	**89.9**	59.8	**63.9**

Table 10: Test set performance when merging T5-large models on seven NLP tasks. Please refer to Section 5.1 for experimental details.

Task(→) / Method(↓)	Validation	Average	Test Set Performance						
			paws	qasc	quartz	story_cloze	wiki_qa	winogrande	wsc
Zeroshot	-	53.1	58.2	54.2	54.1	54.3	70.9	49.2	63.9
Fine-tuned	-	88.9	94.5	98.3	88.5	91.4	96.2	74.5	79.2
Multitask	-	88.1	94.2	98.5	89.3	92	95.4	73.5	73.6
Averaging[ICML22] [83]	✗	54.7	57.2	26.4	71.4	54.8	86.6	50.2	36.1
Task Arithmetic[ICLR23] [26]	✗	73.6	69.7	83.6	58.3	77.4	94.4	59.3	72.2
Ties-Merging[NeurIPS23] [86]	✗	71.7	71.2	97.1	74.2	74.9	73.3	62.9	48.6
PCB-MERGING (ours)	✗	77.1	78.1	98	**75.4**	77.7	89.1	64.6	56.9
Fisher Merging[NeurIPS22] [43]	✓	68.7	68.4	83	65.5	62.4	94.1	58.2	49.2
RegMean[ICLR23] [27]	✓	79.8	**83.9**	97.2	73.2	82.6	94.1	63.2	64.4
Task Arithmetic[ICLR23] [26]	✓	80.2	77.6	96.6	75.1	**85.6**	93.8	61.8	70.8
Ties-Merging[NeurIPS23] [86]	✓	80.3	78.2	97.5	72.8	83.7	**94.5**	64.5	70.8
PCB-MERGING (ours)	✓	**82.1**	82.0	**98.4**	72.2	**85.6**	94.0	**67.5**	**75.0**

Table 11: Test set performance when merging $(IA)^3$ models on eleven tasks. Please refer to Section 5.1 for experimental details.

Task(→) / Method(↓)	Validation	Average	Natural Language Inference					Sentence Completion			Co-reference		WSD
			RTE	CB	ANLI1	ANLI2	ANLI3	COPA	Hella.	Story.	WSC	Wino.	WiC
Zeroshot	-	53.1	58.2	54.2	35.5	34.4	34.4	75.0	39.2	86.5	63.9	51.2	51.9
Fine-Tuned	-	71.4	82.7	95.8	70.4	46.5	53.0	85.3	44.4	95.0	65.3	75.1	71.7
Averaging[ICML22] [83]	-	57.9	81.2	58.3	43.3	39.1	40.0	80.9	40.1	92.4	52.8	53.8	55.0
Task Arithmetic[ICLR23] [26]	✗	59.2	76.5	79.2	59.8	47.5	48.2	66.2	31.4	81.5	51.4	57.7	51.6
TIES-Merging[NeurIPS23] [86]	✗	64.9	81.2	**87.5**	58.1	46.5	47.4	80.2	42.6	91.1	58.3	60.8	59.9
PCB-MERGING (ours)	✗	66.1	**85.9**	83.3	64.2	47.8	45.9	82.4	**42.7**	91.2	**63.9**	61.9	57.1
Fisher Merging[NeurIPS22] [43]	✓	62.2	83.3	83.3	45.9	41.0	42.2	83.1	42.2	94.1	58.3	56.7	54.2
RegMean[ICLR23] [27]	✓	58	81.2	58.3	43.3	39.2	40.2	80.9	40.1	92.5	53.5	53.8	55
Task Arithmetic[ICLR23] [26]	✓	63.9	74.1	83.3	60.8	49.4	50.0	87.5	41.5	**95.3**	49.3	62.8	49.1
TIES-Merging[NeurIPS23] [86]	✓	66.8	78.6	**87.5**	66.6	**51.3**	**51.5**	81.7	43.2	90.9	57.6	67.0	58.4
PCB-MERGING (ours)	✓	**68.1**	80.0	83.3	**67.1**	51.1	49.6	**88.3**	**42.7**	92.8	61.8	**67.6**	**64.7**

Table 12: Test set performance when merging ViT-B/32 models on 8 vision tasks. Please refer to Section 5.1 for experimental details.

Task(→) / Method(↓)	Validation	Average	Test Set Performance							
			SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD
Individual	-	90.5	75.3	77.7	96.1	99.7	97.5	98.7	99.7	79.4
Multitask	-	88.9	74.4	77.9	98.2	98.9	99.5	93.9	72.9	95.8
Averaging[ICML22] [83]	✗	65.8	65.3	63.4	71.4	71.7	64.2	52.8	87.5	50.1
Task Arithmetic[ICLR23] [26]	✗	60.4	36.7	41	53.8	64.4	80.6	66	98.1	42.5
Ties-Merging[NeurIPS23] [86]	✗	72.4	59.8	58.6	70.7	79.7	86.2	72.1	**98.3**	54.2
PCB-MERGING (ours)	✗	75.9	65.8	64.4	78.1	**81.1**	84.9	**77.1**	98.0	58.4
Fisher Merging[NeurIPS22] [43]	✓	68.3	**68.6**	**69.2**	70.7	66.4	72.9	51.1	87.9	**59.9**
RegMean[ICLR23] [27]	✓	71.8	65.3	63.5	75.6	78.6	78.1	67.4	93.7	52
Task Arithmetic[ICLR23] [26]	✓	70.1	63.8	62.1	72	77.6	74.4	65.1	94	52.2
Ties-Merging[NeurIPS23] [86]	✓	73.6	64.8	62.9	74.3	78.9	83.1	71.4	97.6	56.2
PCB-MERGING (ours)	✓	**76.3**	66.7	65.5	**78.5**	79.3	**86.4**	**77.1**	98.2	59.1

Table 13: Test set performance when merging ViT-L/14 models on 8 vision tasks. Please refer to Section 5.1 for experimental details.

Task(→) / Method(↓)	Validation	Average	Test Set Performance							
			SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD
Fine-tuned	-	94.2	82.3	92.4	97.4	100	98.1	99.2	99.7	84.1
Multitask	-	93.5	90.6	84.4	99.2	99.1	99.6	96.3	80.8	97.6
Averaging[ICML22] [83]	✗	79.6	72.1	81.6	82.6	91.9	78.2	70.7	97.1	62.8
Task Arithmetic[ICLR23] [26]	✗	83.3	72.5	79.2	84.5	90.6	89.2	86.5	**99.1**	64.3
Ties-Merging[NeurIPS23] [86]	✗	86	76.5	85	89.3	95.7	90.3	83.3	99	68.8
PCB-MERGING (ours)	✗	86.9	75.8	86	89.2	96	88	90.9	99.1	70
Fisher Merging[NeurIPS22] [43]	✓	82.2	69.2	**88.6**	87.5	93.5	80.6	74.8	93.3	70
RegMean[ICLR23] [27]	✓	83.7	73.3	81.8	86.1	97	88	84.2	98.5	60.8
Task Arithmetic[ICLR23] [26]	✓	84.5	74.1	82.1	86.7	93.8	87.9	86.8	98.9	65.6
Ties-Merging[NeurIPS23] [86]	✓	86	76.5	85	**89.4**	95.9	**90.3**	83.3	99	68.8
PCB-MERGING (ours)	✓	**87.5**	**76.8**	86.2	**89.4**	**96.5**	88.3	**91**	98.6	**73.6**

Additionally, we present the results of merging vision tasks using radar charts for a more intuitive comparison of performance across each task, as shown in Fig. 9. The previous baseline methods show unstable performance, with poor results in some tasks. In contrast, our method is more robust, achieving near-best performance across all tasks.

We also present task-level results of cross-domain merging experiments, as introduced in Section 5.2. Firstly, we fine-tuned five distinct domain-specific models for Emotion Classification and then employed different model merging methods to obtain a single model. For models with an encoder-only architecture, we used the same shared classification head initialization during merging. We tested the performance of the merged model on the original five domains and its generalization on

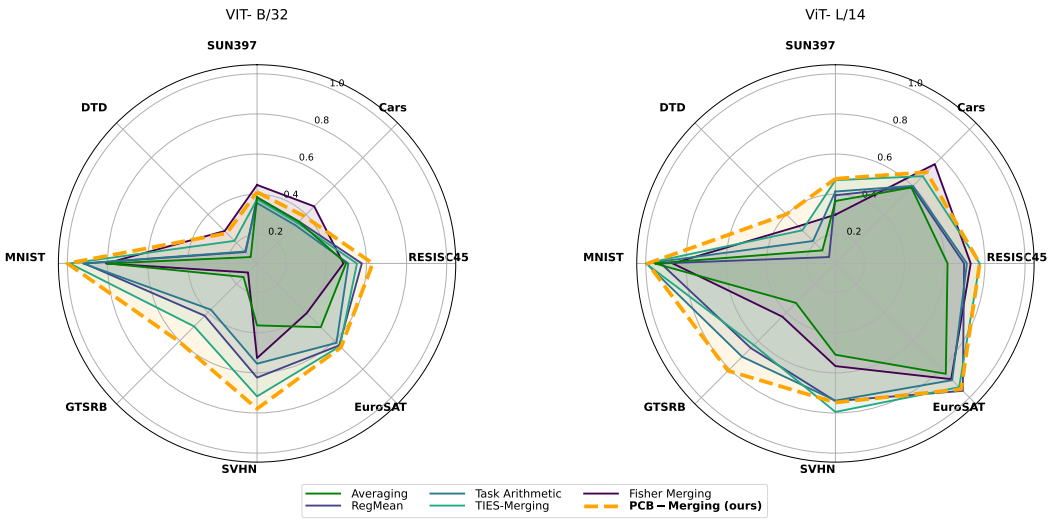

Figure 9: Test set performance when merging ViT-B/32 and ViT-L/14 models on eight image classification tasks.

Table 14: Out-of-distribution performance across six held-out tasks after merging the checkpoints of T5-base and T5-large models from seven NLP tasks. Please refer to Section 5.1 for experimental details.

Task(→) / Method(↓)	model	Average	Question Answering			WSD	Sentence Completion	
			cosmos_qa	social_iqa	quail	wic	copa	h-swag
Pretrained	T5-base	31.1	21.9	18.8	24.1	65.6	43.8	12.5
Averaging[ICML22] [83]		31.7	21.9	21.9	24.6	**68.8**	37.5	15.6
Fisher Merging[NeurIPS22] [43]		33.8	15.6	21.9	24.9	65.6	53.1	21.9
Task Arithmetic[ICLR23] [26]		31.9	15.6	**31.2**	25.7	28.1	**68.8**	21.9
RegMean[ICLR23] [27]		34.3	23.1	28.1	24.9	48.4	62.5	18.8
TIES-Merging[NeurIPS23] [86]		35.3	21.9	25	25.7	50	65.6	23.8
PCB-MERGING (ours)		**37.2**	**23.6**	29.2	**26.6**	51.9	67.1	**24.8**
Pretrained	T5-large	27.6	21.9	21.9	24.9	28.1	56.2	12.5
Averaging[ICML22] [83]		30.4	31.2	25	26.3	31.2	59.4	9.4
Fisher Merging[NeurIPS22] [43]		32	34.4	25	26.1	40.6	56.2	9.4
Task Arithmetic[ICLR23] [26]		33.3	21.9	34.4	24.6	40.6	59.4	18.8
RegMean[ICLR23] [27]		36	**34.4**	28.1	25.3	**62.5**	50	15.6
TIES-Merging[NeurIPS23] [86]		40.4	31.2	43.8	26.6	59.4	59.4	21.9
PCB-MERGING (ours)		**42.5**	33.6	**45.8**	**29.6**	62.2	**59.2**	**24.6**

unseen datasets from five other domains. For more dataset details, please refer to App. D. To ensure the reliability of the results, we fine-tuned the models five times with different random seeds and reported the average performance for these runs, as shown in Tab. 15.

Table 15: In domain and Out of domain performance when merging Roberta-base models on 5 emotion datasets. Please refer to Section 5.2 for experimental details.

Dataset(→)	In Domain						Out of Domain					
Method(↓)	Average	Dialy.	Crowd.	TEC	Tales	ISEAR	Average	Emoint	SSEC	Elect.	Ground.	Affec.
Fine-Tuned	51.38	49.3	28.9	56.4	49.2	73.1	-					
Averaging[ICML22] [83]	23.2	29.9	16.6	17.0	25.2	27.1	11.6	27.8	5.2	6.5	14.0	4.3
Fisher Merging[NeurIPS22] [43]	26.1	29.8	**25.9**	19.5	26.2	29.0	16.2	32.7	10.7	12.0	14.8	10.9
RegMean[ICLR23] [27]	34.2	**33.1**	20.7	34.1	35.0	48.3	21.3	43.	15.4	**13.7**	**20. 0**	14.6
TIES-Merging[NeurIPS23] [86]	34.5	32.2	20.6	35.5	35.1	49.3	21.5	43.4	16.1	13.3	19.7	15.0
PCB-MERGING (ours)	**35.6**	32.1	21.2	**37.4**	**36.0**	**51.2**	**22.2**	**44.2**	**17.5**	13.5	19.7	**16.1**

D Dataset details

This section provides a detailed dataset description.

Merging NLP Tasks Following TIES-Merging [86], we choose seven datasets for merging NLP models: question answering (QASC [29], WikiQA [88], and QuaRTz [75]), paraphrase identification (PAWS [93]), sentence completion (Story Cloze [67]), and coreference resolution (Winogrande [62] and WSC [34]).

Merging PEFT Models Following TIES-Merging [86], we use eleven datasets including sentence completion (COPA [58], H-SWAG [91], and Story Cloze [67] datasets), natural language inference (ANLI [49], CB [42], and RTE [17]), coreference resolution (WSC [34] and Winogrande [62]), and word sense disambiguation (WiC [53]).

Merging Vision Tasks Following Task Arithmetic [26], we study multi-task model merging on eight image classification datasets below. Stanford Cars [32] is a car classification dataset consisting of 196 classes of cars. DTD [9] is a texture classification dataset comprising 47 classes. EuroSAT [20] comprises 10 classes of geo-referenced satellite images. GTSRB [71] includes 43 classes of traffic signs. MNIST [33] features grayscale images of handwritten digits across 10 classes. RESISC45 [7] encompasses 45 classes of remote sensing image scenes. SUN397 [84] consists of 397 classes of scene images. Lastly, SVHN [48] encompasses 10 classes of real-world digital classification images.

Merging LLMs

- **CMMLU** [35] is a comprehensive Chinese evaluation benchmark specifically designed to assess language models' knowledge and reasoning abilities in a Chinese context. It covers 67 topics ranging from basic subjects to advanced professional levels.
- **GSM8K** [10] is a collection of 8.5K high-quality, linguistically varied math word problems from grade school, crafted by skilled human authors. The solutions predominantly require executing a series of basic arithmetic operations ($+$, $-$, $\times$, $\div$) to derive the final answer.
- **HumanEval** [6] is a dataset for evaluating code generation ability, containing 164 manually crafted programming problems covering aspects such as language understanding, reasoning, algorithms, and simple mathematics.

Table 16: Statistics of in domain and out-of-domain emotion classification datasets.

	Train	Dev	Test
In-domain			
DialyDialog	72,085	10,298	20,596
CrowdFlower	27,818	3,974	7,948
TEC	14,735	2,105	4,211
Tales-Emotion	10,339	1,477	2,955
ISEAR	5,366	766	1,534
Out-of-domain			
Emoint			7,102
SSEC			4,868
ElectoralTweets			4,056
GroundedEmotions			2,585
AffectiveText			1,250

Out of Domain Generalilzation The average performance is reported over the following tasks and datasets: Cosmos QA [24], Social IQA [64], and QuAIL [59] for question answering; WiC [53] for word sense disambiguation; and COPA [58], and H-SWAG [91] for sentence completion.

Cross-Domain Merging In order to investigate the performance of the sentiment classification task, following RegMean [27], we selected a diverse and challenging set of datasets. Among them, DailyDialogs [38], CrowdFlower, TEC [46], Tales-Emotion [2], and ISEAR [65] is utilized to train domain-specific model. For acessing OOD generalization performance, we use Emoint [45], SSEC [66], ElectoralTweets [47], GroundedEmotions [40], and AffectiveText [73]. For OOD evaluation, we focus exclusively on the fundamental emotions: anger, disgust, fear, joy, sadness, and surprise. A detailed overview of the datasets and statistics is provided in Tab. 16.

Cross-Training Configurations Merging We study four GLUE benchmark text classification datasets [79]. (1) MRPC [11]: Sentence pairs labeled for semantic equivalence; (2) RTE [17]: Sentence pairs for entailment prediction; (3) CoLA [81]: Sentences labeled for grammaticality; (4) SST-2 [70]: Sentences labeled for sentiment.

E Baseline details

This section provides a detailed baseline description. Our experiments encompass seven comparison methods:

- **Individual** means that each task uses an independent fine-tuned model, which has no interference between tasks, but cannot perform multiple tasks simultaneously.
- **Traditional MTL** collects the original training data of all tasks together to train a multi-task model. It can be used as a reference *upper bound* for model merging work.
- **Weight Averaging** is the simplest method of model merging, which directly averages the parameters of multiple models using $\theta_m = \sum_{t=1}^{n} \theta_t / n$, calculating the element-wise mean of all individual models. It can be used as a *lower bound* for model merging. [8, 83].
- **Fisher Merging** [43] calculates the Fisher information matrix [15] $\hat{F}_t = \mathbb{E}_{x \sim D_t} \mathbb{E}_{y \sim p_{\theta_t}(y|x)} \nabla_{\theta_t} (\log p_{\theta_t}(y|x_t))^2$ to measure the importance of each parameter when merging models for task t, where and model merging is performed according to the guidance of this importance.
- **RegMean** [27] imposes a constraint when merging models, that is, the L_2 distance between the merged model's and the individual models' activations. It computes a least-squares solution as $\theta_m = (\sum_{t=1}^{n} X_t^T X_t)^{-1} \sum_{t=1}^{n} (X_t^T X_t \theta_t)$, where X_t is the input activation of the corresponding layer.
- **Task Arithmetic** [26] first defines the concept of "task vectors" and merges these vectors into a pre-trained model to execute multi-task learning. The model is produced by scaling and adding the task vectors to the initial model as $\theta_m = \theta_{\text{init}} + \lambda * \sum_{t=1}^{n} \tau_t$.
- **Ties-Merging** [86] further solves the task conflict problem in Task Arithmetic [26]. It eliminates redundant parameters and resolves symbol conflicts through three steps: Trim, Elect Sign, and Disjoint Merge.
- **AdaMerging** automatically learns a merging coefficient for each layer of each task vector in Task Arithmetic [26].
- **LoraHub** [23] employs Low-rank Adaptations to dynamically combine task-specific modules for cross-task generalization, and adapts to new tasks by configuring $\theta' = \sum_{k=1}^{K} w_k \cdot \theta_k$.
- **DARE** [90] sets the majority of delta parameters to zero and rescale the rest by $\theta' = \theta \cdot (1/(1-p))$ where p is the proportion of delta parameters dropped, therefore efficiently reduces parameter redundancy.

F Implementation details

F.1 Computational Resources and Runtimes

Our experiments were conducted on Nvidia A6000 GPUs with 48GB of RAM. Depending on the dataset size, fine-tuning the T5-Base and T5-Large models for single tasks took between 15 minutes and 2 hours, while fine-tuning the multitask checkpoint took around eight hours. The fine-tuned $(IA)^3$ models were provided by Yadav et al. [86].[4]. We also used vision models ViT-B/32 and ViT-L/14 as provided by Ilharco et al. [26].[5].

Merge experiments were highly efficient, with evaluations for RoBerta-base, T5-Base, T5-Large, ViT-B/32, and ViT-L/14 models taking less than 2 minutes. However, two specific experiments required more time: (1) Evaluating $(IA)^3$ models took about one hour for 11 datasets due to the need to use multiple templates from prompt sources and compute median results across them. (2) Validation on LLMs (LLaMa2) was also slow, usually requiring about 40 minutes for evaluating 3 datasets.

F.2 Training details

Cross-Task Merging We trained the T5-base and T5-large models for up to 75,000 steps, using an effective training batch size of 1024 and a learning rate of 0.0001. To prevent overfitting, we implemented an early stopping mechanism with a patience of 5. Training was conducted in bfloat16 to

[4] https://github.com/prateeky2806/ties-merging

[5] https://github.com/mlfoundations/task_vectors#checkpoints

conserve GPU memory, with a maximum sequence length of 128 tokens. For the PEFT configuration of the $(IA)^3$ approach on the T0-3B model, we adjusted the parameters accordingly. The training batch size was set at 16, and the evaluation batch size was 32, while keeping the learning rate at 0.0001. Given the increased complexity, we extended the early stopping patience to 10. No learning rate scheduler or weight decay was used in any of our training processes. For large language models, we directly utilized the fine-tuned checkpoints provided by Huggingface[6].

Cross-Domain Merging We performed fine-tuning of the RoBERTa-base model starting with an initial learning rate of 1e-5, and for the T5-base model, we used an initial learning rate of 1e-4. We applied the AdamW optimizer consistently across all experiments. The learning rate was set to gradually increase during the first 6% of training steps and then linearly decreased to zero. The models were trained with a batch size of 16 over 30 epochs for the task of emotion classification. We assessed model performance at the end of each epoch and, upon completing the training, resumed from the best-performing checkpoint.

Cross-Training Configurations Merging When merging multiple checkpoints of the same task, each model is fine-tuned 10 times on each dataset using a random hyperparameter search. The learning rate is randomly selected in log space from $[10^{-6}, 10^{-3}]$, the batch size from $\{8, 16, 32, 64\}$, and the number of epochs from $\{2, 3, 5\}$. Evaluation occurs once at the end of training without early stopping. We use a maximum sequence length of 128 tokens and train the models using the Adam optimizer [30], with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. Training includes gradient clipping at 1.0, no weight decay, and a learning rate that linearly decays to zero by the end of the process.

F.3 Hyper-parameter settings

Given the sensitivity of task vector-based model merging methods to hyperparameters, we present the optimal values of λ and r as determined in our experiments, as shown in Tab. 17. For Task Arithmetic, we conduct a search over λ ranging from 0.2 to 1.5 with a step size of 0.1. For TIES-Merging and PCB-MERGING, we search over mask ratios r in {0.05, 0.1, 0.2}, and λ ranging from 0.8 to 2.5 with a step size of 0.1.

Table 17: Optimal λ and mask ratio r for cross-task merging

Task (→)	7 **NLP** Tasks		11 **PEFT** Tasks	3 **LLM** Tasks	8 **Vision** Tasks	
Method (↓)	T5-Base	T5-Large	$(IA)^3$	LLaMa2	ViT-B/32	ViT-L/14
Task Arithmetic[ICLR23] [26] [λ]	0.4	0.5	0.5	0.3	0.3	0.3
Ties-Merging[NeurIPS23] [86] [λ, r]	[1.7, 0.1]	[2.4, 0.05]	[1.7, 0.1]	[1.0, 0.1]	[1.0, 0.1]	[1.1, 0.05]
PCB-MERGING (ours) [λ, r]	[1.9, 0.05]	[2.2, 0.05]	[1.8, 0.1]	[0.9, 0.1]	[1.2, 0.05]	[1.2, 0.05]

[6] https://huggingface.co/