FREE-Merging: Fourier Transform for Efficient Model Merging

Supplementary Material

A. Notations

We first list the notations for key concepts in our paper.

Table 10. Notations.

Notations	Descriptions
f_θ	A neural network with parameters θ.
θ_{pre}	Pre-trained model parameters.
θ_k	Fine-tuning parameters for task k.
θ_m	Model merging backbone network.
v_k	Task vector for task k, $v_k = \theta_k - \theta_{pre}$.
$M(v_k, d)$	The top 100d% parameters with the most significant changes corresponding to v_k.
μ_k	The rescaling coefficient of the expert of task k
e_k	Lightweight task expert for task k, $e_k = \mu_k M(v_k, d)$.
θ_*	The backbone with the corresponding expert.
$G(\cdot)$	Transformation operations on parameters.
λ_k	Merging coefficient of the model parameters for task k.
$x \in R^m$	The input of the neural network.
$Y \in R^n$	The output of the neural network.

B. Proof

In this section, we provide a proof for Theorem 5.1. We begin by introducing the notation used in this proof.

B.1. Notations and preliminaries

To illustrate the proof, let's consider two fine-tuned models with parameters θ_A and θ_B, For clarity, we focus on a single-layer MLP (without activation functions), though we will explain at the end how our proof generalizes to all cases. The merged result is given by: $\theta_m = \lambda_1 G(\theta_A) + \lambda_2 G(\theta_B)$, where $G(\cdot)$ represents a transformation operation applied to the parameters. Assume that θ_A corresponds to task input X_A, and θ_B corresponds to task input X_B.

The goal is to prove that, without additional storage or extra training for θ_A and θ_B, simply storing θ_m cannot perfectly retain the capabilities of both θ_A and θ_B.

B.2. Details of proof

Next, we will analyze the cases separately.

Without transformations. Assume that we do not apply transformations to the output of the merged network. the output of the model f_θ can be expressed as:

$$output_f = \theta x \tag{5}$$

Suppose θ_m can perfectly retain the capabilities of both θ_A and θ_B without introducing new knowledge. That is, for $\forall x_A \in X_A, x_B \in X_B$, we have:

$$\theta_m x_A = \theta_A x_A,\ \theta_m x_B = \theta_B x_B. \tag{6}$$

Then we have:

$$[\lambda_1 G(\theta_A) + \lambda_2 G(\theta_B)] x_A = \theta_A x_A, \tag{7}$$

$$[\lambda_1 G(\theta_A) + \lambda_2 G(\theta_B)] x_B = \theta_B x_B. \tag{8}$$

Assume that $\exists\ x_m = \mu_1 x_A + \mu_2 x_B \in X_A$, where $\mu_1, \mu_2 \in \mathbb{N}$. Substituting, we can obtain:

$$\theta_m x_m = \theta_A x_m \tag{9}$$

$$\Rightarrow \theta_m (x_A - x_m) = \theta_A (x_A - x_m) \tag{10}$$

$$\Rightarrow \theta_m [(1-\mu_1) x_A - \mu_2 x_B] \tag{11}$$

$$= \theta_A [(1-\mu_1) x_A - \mu_2 x_B] \tag{12}$$

$$\Rightarrow \mu_2 \theta_m x_B = \mu_2 \theta_A x_B \tag{13}$$

$$\Rightarrow \mu_2 \theta_B x_B = \mu_2 \theta_A x_B \tag{14}$$

Due to:

$$\mu_2 \theta_m x_B = \mu_2 \theta_B x_B \tag{15}$$

Thus, if the Eq. 14 and Eq. 15 holds for $\forall x_A \in X_A, x_B \in X_B$, then $\mu_2 = 0$. Therefore, we have:

$$\forall \mu_2 \neq 0, \nexists\ x_m = \mu_1 x_A + \mu_2 x_B \in X_A \tag{16}$$

Eq. 16 indicates that X_A is not continuous. However, we usually consider the input space of the model to be continuous. Even if it is discontinuous, we expect the model to be robust to slight perturbations in the input.

Thus, the assumption that θ_m can perfectly retain θ_A and θ_B without extra data or training does not hold in this case.

With transformations. Assuming we apply certain transformations to the output of the merged network, the output of the model f_θ can be expressed as:

$$output_f = h(\theta x), \tag{17}$$

where $h(\cdot)$represents a transformation applied to the output.

By the universal approximation theorem, $h(\cdot)$ can be approximated by a two-layer MLP with sigmoid activation. Then, assuming θ_m can perfectly retain the capabilities of both θ_A and θ_B. Then for $\forall x_A \in X_A, x_B \in X_B$, we have:

$$W_2[sigmoid(W_1 \theta_m x_A)] = \theta_A x_A \tag{18}$$

$$W_2[sigmoid(W_1 \theta_m x_B)] = \theta_B x_B. \tag{19}$$

Substituting, we obtain:

$$\frac{W_2}{1 + e^{-W_1 \theta_m x_A}} = \theta_A x_A, \tag{20}$$

$$\frac{W_2}{1 + e^{-W_1 \theta_m x_B}} = \theta_B x_B. \tag{21}$$

Then we have:

$$W_2 = \theta_A x_A (1 + e^{-W_1 \theta_m x_A}) \tag{22}$$
$$= \theta_B x_B (1 + e^{-W_1 \theta_m x_B}). \tag{23}$$

Thus, we can obtain the following equation:

$$\theta_A x_A (e^{W_1 \theta_m x_A} + 1) e^{W_1 \theta_m x_B} \tag{24}$$
$$= \theta_B x_B (e^{W_1 \theta_m x_B} + 1) e^{W 1 \theta_m x_A}. \tag{25}$$

Therefore,

$$(\theta_A x_A - \theta_A x_B)(e^{W_1 \theta_m x_B} e^{W_1 \theta_m x_A} + e^{W_1 \theta_m x_B}) \tag{26}$$
$$= \theta_B x_B (e^{W_1 \theta_m x_B} + e^{W_1 \theta_m x_A}) \tag{27}$$

Therefore, based on Equa. 22 and Equa. 26, since they hold for $\forall x_A \in X_A, x_B \in X_B$, at least one of W_1 or W_2 can be expressed as $\Phi(\theta_A, \theta_B)$, where Φ represents a function.

In other words, only by additionally storing part of the information from θ_A or θ_B and processing θ_m at the output can θ_m perfectly retain the capabilities of both θ_A and θ_B. Therefore, the assumption that θ_m can perfectly retain both θ_A and θ_B does not hold in this case.

Next, we briefly outline the rationale for extending this result to all layers of a neural network. Our assumptions involve only a single layer of the neural network, which already cannot perfectly retain capabilities without introducing new data. Therefore, it is even less feasible for the entire network to achieve this. Thus, this extension is reasonable.

In summary, the proposition is proven.

C. Reproducibility

C.1. Datasets

Merging 8 ViTs. We use ViT-B/32 and ViT-L/14 as pre-trained models, and fine-tune them on 8 image classification datasets (SUN397, Cars, RESISC45, EuroSAT, SHVN, GT-SRB, MNIST, and DTD), then merge the models and test their performance. Configuration details follow [20].

Merging 30 ViTs. We use ViT-B/16 as the pre-trained model and test its merging performance on 30 image datasets. The datasets include MNIST, CIFAR-10, Vegetables, Food-101, Kvasir-v2, Cars, Intel Images, EuroSAT, Weather, Cats and Dogs, MangoLeafBD, Beans, CIFAR-100, GTSRB, SHVN, Dogs, Fashion MNIST, Oxford-IIIT-Pet, Landscape Recognition, Flowers Recognition, STL-10, CUB-200-2011, EMNIST, DTD, RESISC45, SUN397, KenyanFood13, Animal-10N, Garbage Classification, and Fruits-360. These datasets cover a wide range of major image categories. Configuration details follow [20].

Merging Medium-sized Language Models. We use RoBERTa as the pre-trained model, fine-tune it on 8 classification task datasets from GLUE benchmark for model merging, including CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI, and RTE. CoLA is evaluated with the Matthews correlation coefficient, STS-B with the average of the Pearson and Spearman correlation coefficients, and the others by accuracy. Details follow [20].

Merging PEFT Models. In this section, we conduct two experiments. The first uses T0-3B as the pre-trained model and IA^3 as the PEFT method, fine-tuned and merged on RTE, CB, Winogrande, WiC, WSC, COPA, H-SWAG, Story Cloze, and ANLI (R1 to R3). Details can be found in [20]. The second part uses Qwen-14B as the pre-trained model and LoRA as the PEFT method. We fine-tune and merging on three generative tasks: MMLU, TruthfulQA, and BBQ. Configuration details can be found in [29].

Merging Large Language Models. We use LLaMa2-13B as the pre-trained model and apply fine-tuned results from WizardLM, WizardMath, and WizardCoder-Python for instruction following, math, and code tasks, respectively. We perform merging and test the model on three datasets: AlpacaEval (instruction following task), GSM8K (math task), and MBPP (code generation task). Details follow [53].

Merging Multi-Modal Models. We use BEiT3 as the pre-trained model, fine-tune it on five multi-modal datasets, and test model merging performance. The datasets are ImageNet-1k (Image Classification), VQAv2 (Visual Question Answering), NLVR2 (Visual Reasoning), COCO Captioning (Image Captioning), and COCO Retrieval (Image-Text Retrieval). COCO Captioning is evaluated using BLEU4, CIDEr, METEOR, and ROUGE-L, while the other tasks are measured by accuracy. Details follow [20].

C.2. Baselines

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

Traditional MTL. Train a model on data from all tasks, which incurs high training costs and data privacy issues.

Weight Averaging. Average the parameters for merging. The method is simple, but it faces great performance loss.

Fisher Merging [33]. Use Fisher information matrices to assess parameter importance and determine merging coefficients. However, this requires calculating the matrix for each model, demanding high computational resources, making it unsuitable for edge deployment.

Task Arithmetic [21]. Define task vectors as the merging target. For task k, the task vector is defined as $v_k = \theta_k - \theta_{pre}$, where θ_{pre} is the pre-trained model parameters, and θ_k is the fine-tuned parameters for task k. The merging process can be represented as $\theta_m = \theta_{pre} + \lambda \sum_{i=1}^{K} v_i$, where λ is the merging coefficient. This method suffers significant performance degradation due to unaddressed task conflicts.

Ties-Merging [48]. Attempts to resolve parameter conflicts during model merging by eliminating redundancy and sign conflicts. However, resolving parameter conflicts is insufficient to address task conflicts, resulting in performance loss.

Breadcrumbs [7]. Discards parameters with the largest and smallest absolute values as redundant, negatively impacting model merging. This approach is simple, but it shows a significant performance decline on certain tasks.

PCB-Merging [12]. Uses internal balancing to measure parameter importance within tasks and mutual balancing to assess parameter similarity across tasks, discarding redundant parameters and adjusting merging coefficients. This approach requires considerable computational resources, whereas our FR-Merging approach requires less computational power while yielding better results.

RegMean [23]. A weighted merging model based on a closed-form solution for the merging problem, aiming to maximize the similarity between the merged model and each pre-merged model on the same input. Requires inner product data from training. Since this method relies on data information, which is typically not provided by the model provider, its applicability is limited to specific scenarios.

AdaMerging [52]. Uses an unsupervised approach to learn the merging coefficient for each task vector or layer. AdaMerging++ additionally applies Ties-Merging before calculating the merging coefficient. This method is limited to classification tasks and requires certain training resources, making it unsuitable for edge deployment.

DARE [53]. DARE randomly discards a large portion of task vector parameters before merging, potentially reducing parameter interference among models. This method is simple, but due to the lack of further optimization in the merging, it suffers from significant performance degradation.

EMR-Merging [20]. Proposes retaining a mask matrix and rescale parameters for each model. During inference, the mask matrix and rescaling parameters are selected to recover performance. This method cannot further improve performance because it does not optimize the merged backbone. Additionally, since a mask must be stored for each task, it requires considerable storage space. In contrast, our method optimizes these aspects simultaneously.

Twin-Merging [29]. Proposes saving an expert module extracted by SVD from the original parameters for each task, which is dynamically added during inference. This method does not optimize the merged backbone, resulting in suboptimal performance. It requires storing a large number of parameters when constructing task experts. In contrast, our proposed method optimizes both aspects, making it suitable for resource-constrained edge deployment scenarios.

C.3. Details

In this section, we discuss the details of the experiments. All experiments, except for the speed tests, are conducted on eight NVIDIA RTX-3090 GPUs. The inference speed tests are performed using a single RTX-3090 GPU.

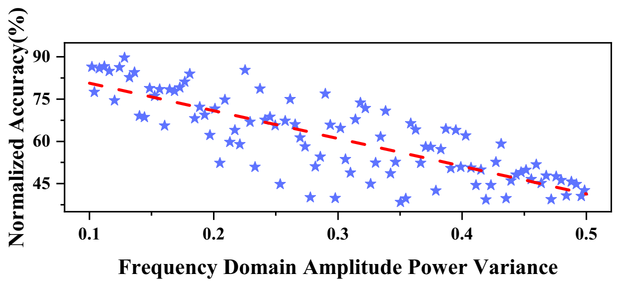

Figure 6. The relationship between the merging performance of ViT-B-32 and the variance of frequency domain amplitude power. Normalized accuracy is the ratio of merged model performance to fine-tuned model performance on the same task.

D. More Discussions and Future Directions

In this section, we first provide a more detailed discussion of task interference in the frequency domain in D.1. Next, we present the experimental results of FR-Merging on language models in D.2 and D.3. Additionally, we discuss the effectiveness and construction of the expert module in D.4 and D.5, as well as the role of the router in D.6. We also explore the selection of the cutoff frequency in D.7 and explored the implications of forgetting model-specific capabilities in D.8. Finally, we conclude with a discussion of future work in D.9.

D.1. Task Interference in Frequency Domain

In this section, we further discuss the task interference presented in Sec. 4 and provide visualization results.

The first point to address is why we consider the difference in low-frequency amplitude in the frequency domain as the cause of task interference. The amplitude in the frequency domain represents the strength of a signal. During the merging process, if one signal is strong while the other is weak, the weaker signal is likely to be overshadowed by the stronger one, regardless of their directional relationship. This manifests in the merged model as the near-complete loss of performance from one of the fine-tuned models. Therefore, a significant difference in the low-frequency amplitude will indeed lead to task interference during merging.

In Sec. 4, we quantify interference using the mean variance, observing a strong negative correlation with merged model performance on ViT-B/32. Here, normalized performance refers to the ratio of the performance of merged model on a task to that of the fine-tuned model on the same task. This negative correlation serves as strong evidence of task interference in the frequency domain.

Next, we verify whether this negative correlation also holds for language models. We choose Flan-T5 [6] as the base model and conduct experiments on eight GLUE datasets. As shown in Fig. 6, the negative correlation is observed in language models, demonstrating that task interference in the frequency domain is a general phenomenon.

Finally, we examine whether low-frequency task interference is widespread across different models. In Fig. 7 and Fig. 8, we visualize the frequency distribution of fine-tuned parameters for ViT-B/32 and Flan-T5 [6] on their respective tasks. The results show that task interference in the low-frequency region is consistently present across various models. Therefore, our hypothesis is well-supported. Meanwhile, we observe an interesting phenomenon: in some layers, frequency-domain amplitude clustering occurs, meaning that certain models have very similar amplitude distributions. This may be related to the training path during the finetuning process, and we will further investigate this phenomenon in future research.

D.2. Expanded Loss Landscape Analysis

In this section, we attempt to analyze why our proposed FR-Merging outperforms existing approaches from the perspective of the loss landscape. A loss landscape comparison with two popular sparse methods, Top-K retention and DARE, is shown in Fig. 9 (a-c). Our proposed FR-Merging method brings the two fine-tuned models closer in the loss landscape for the same task, making their linear combination fall closer to the loss basin of the target task, thereby achieving better performance. This aligns with our analysis from the perspective of task interference and is also consistent with the performance improvements in experiments. Therefore, our proposed FR-Merging method offers significant advantages over current approaches.

D.3. Fourier Transform on Language Models

In Sec. 5.2, we analyze the impact of high-pass filtering on the generalization ability of fine-tuned models and their performance on corresponding tasks using ViT-B/32. In this part, we extend the analysis to language models, further demonstrating the effectiveness of high-pass filtering. Fig. 9(d) shows the generalization test of RoBERTa on the GLUE datasets, where filtering out low-frequency information also enhances the generalization ability of fine-tuned language models. This improvement is also reflected in the merging performance improvement for language models. However, compared to the effect on language models, high-pass filtering has a more pronounced impact on ViT. This may be attributed to the nature of image data, which inherently possesses certain high-frequency and low-frequency properties. Nevertheless, in summary, FR-Merging, based on high-pass filtering, is a generalizable, efficient, and high-performing model merging approach.

D.4. Role of Task Experts

In this section, we demonstrate that introducing additional information can potentially mitigate task conflicts.

We use ViT-B/32 as the pre-trained model and fine-tune it on 8 image classification tasks, following the setup in [21]. To prevent the impact of parameter erasure, we select only 5% of non-overlapping parameters for each task. As a result, after this selection, only task conflicts affect the model merging process. Merging resulted in a significant performance drop, indicating the presence of task conflicts. However, when we double task-specific vectors for each task during inference, performance improves substantially. This suggests that introducing additional information for each task has the potential to resolve task conflicts.

Table 11. Using ViT-B/32 as the pre-trained model, we fine-tune on eight classification tasks, selecting 5% non-overlapping parameters per task. Performance improves significantly by adding task-specific parameters during inference.

	Avg acc(%)	Norm acc(%)
Fine-tuned	90.5	100.0
No Overlap 5% params.	89.91	99.34
No Overlap Merging	67.31	74.37
+ Task Knowledge	84.55	93.42

D.5. Task Experts Construction Analysis

In this section, we analyze the construction of experts from two aspects: the parameter selection and rescaling.

Top-K Selection. We discuss the rationale behind using the parameters with the largest changes during the fine-tuning to approximate task-specific information introduced by fine-tuning, thereby constructing task experts.By comparing L1 distances of model parameters in the frequency domain, with smaller distances indicating better approximation, we assess the effectiveness of different methods.

Based on the analysis in Sec 5.3, low-frequency information contains more task-specific information but is discarded during model merging in our proposed FR-Merging. Therefore, task experts should compensate for this missing information. First, we briefly explain why approximation methods are preferred over directly saving low-frequency information. If low-frequency data is directly saved, ensuring lightweight storage requires saving frequency-domain data instead of inverse-transformed results. This would necessitate performing inverse transformations during expert invocation, significantly increasing the inference time. Considering the constraints of time and storage in edge-side deployment, a more efficient approximation method is needed.

Next, we compare the extent to which different approximation methods approximate low-frequency information to assess their suitability for constructing task experts. In Fig 10, we present the effects of approximating using the top 5% largest fine-tuned parameters and random sampling. It can be observed that using the parameters with the largest changes during the fine-tuning closely approximates the frequency-domain characteristics of low-frequency in-

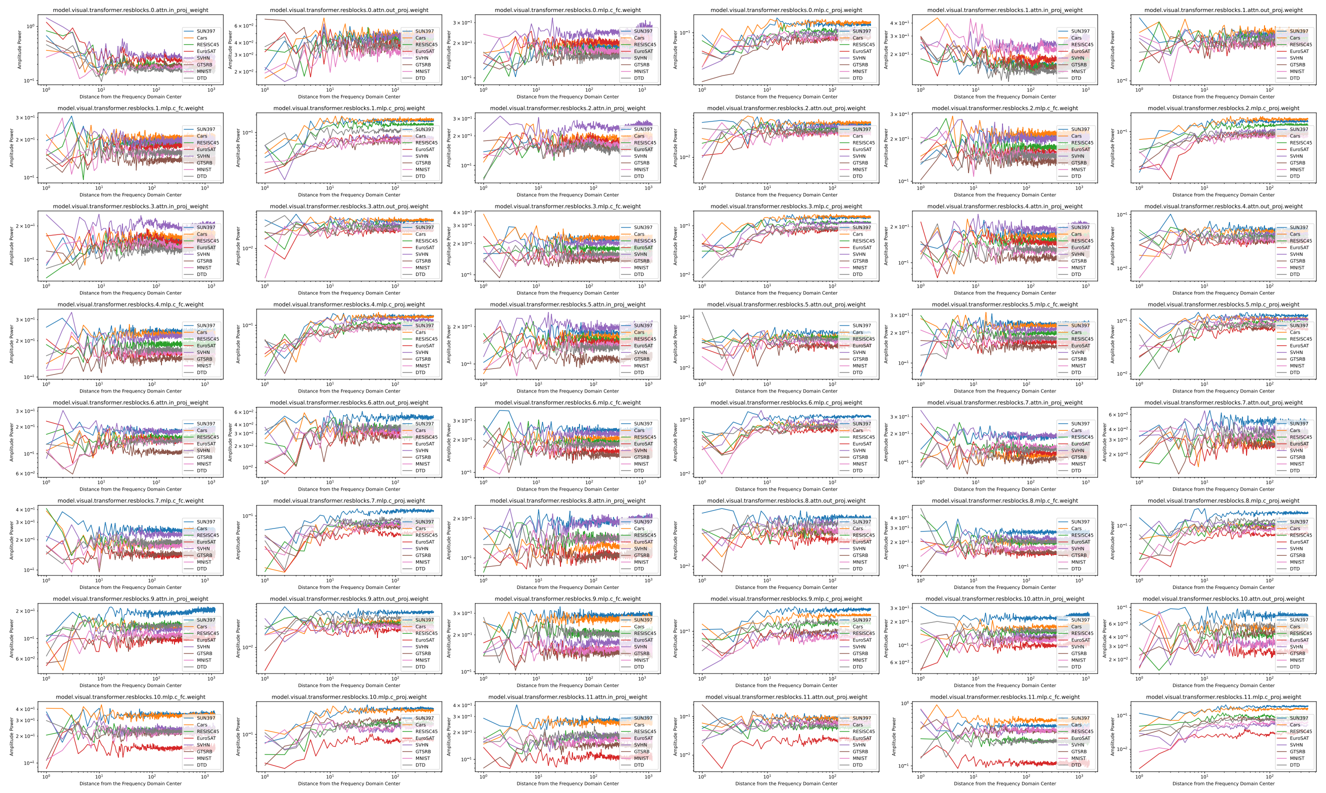

Figure 7. The parameter frequency domain distribution of ViT-B-32 fine-tuning results on eight image tasks.

Table 12. Ablation of scaling for experts on 8 visual tasks.

	ViT-B/32	ViT-B/16	ViT-L/14
w/o scaling	0.5314	0.5323	0.6090
w/ scaling	**0.8484**	**0.8486**	**0.8895**

Table 13. Ablation study of router training.

Training Data Size	0%	0.1%	0.5%	1%	2%	5%
ViT-B/32	78.1	87.0	88.2	89.7	90.0	90.1
ViT-L/14	88.3	92.1	93.0	93.7	93.9	94.1

formation, strongly supporting the rationale for preserving task-specific information by saving the parameters with the largest changes during the fine-tuning. Additionally, the high similarity between low-frequency information and the original information confirms our hypothesis: low-frequency information contains more task-specific information introduced during fine-tuning.

In summary, using the parameters with the largest changes during fine-tuning as task experts to offset task-specific information lost during merging is reasonable. Furthermore, as validated in Sec 6.5, employing our proposed rescaling method yields even better results. Thus, the expert construction method proposed in this paper is justified.

Rescaling. We then discuss the necessity of rescaling. Since we retain only a small portion of the parameters during expert extraction, it alters the mean and variance of parameters, similar to dropout. Our scaling method considers both the mean and the backbone to preserve the similarity between the expert and original model outputs. Table 12 shows the CKA similarity between expert outputs and original outputs before and after scaling, with higher values indicating better expert performance. Scaling can be found to greatly improve performance compared to not applying it.

D.6. Role of Router

In this section, we discuss why a router is necessary for dynamically assigning input data to the appropriate expert. First, let's review how the router works: when a new data arrives, the router determines which existing expert is best suited to handle it. For example, if we have experts for mathematics and physics, and a math problem appears, the router will automatically route it to the mathematics expert. The mathematics expert will then collaborate with the merged backbone to solve the problem. Manually assigning each input sample to the correct task is impractical, and in real-world applications, we cannot expect users to know in advance which expert is best suited for each piece of data. Therefore, having a router to automate this process is both crucial and aligned with real-world usage scenarios.

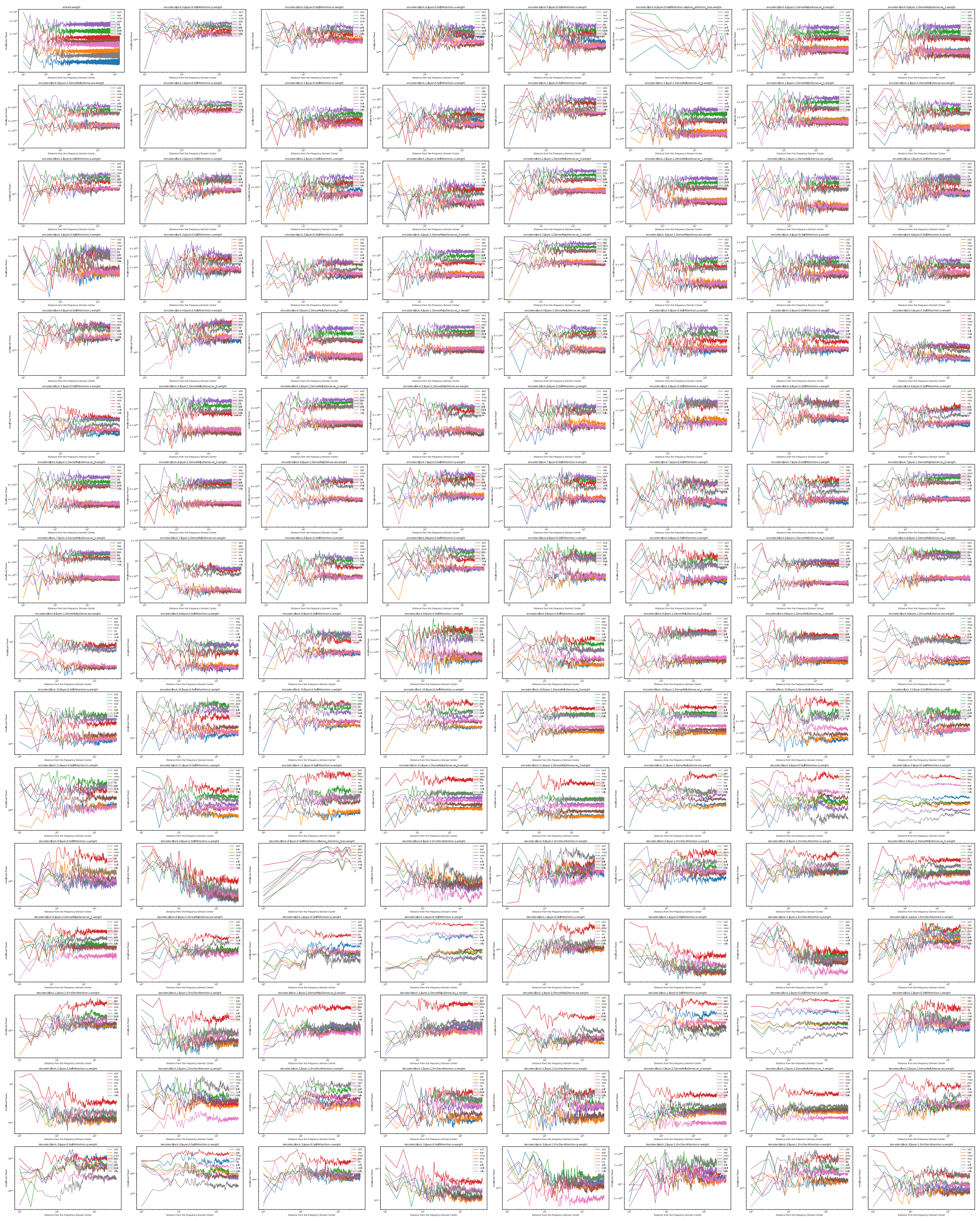
Figure 8. The parameter frequency domain distribution of Flan-T5 fine-tuning results on eight GLUE tasks.

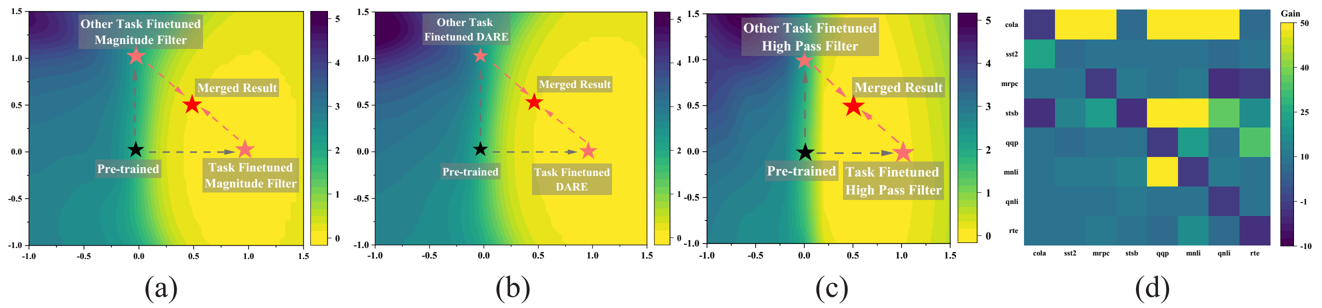

Figure 9. a) Merge by preserving the top 20% of parameters; b) Merge using the DARE method; c) FR-Merging. **a-c** complete on ViT/B-32 and test on 8 visual tasks. d) High-pass filtering improves performance over the unfiltered version on RoBERTa and test on GLUE.

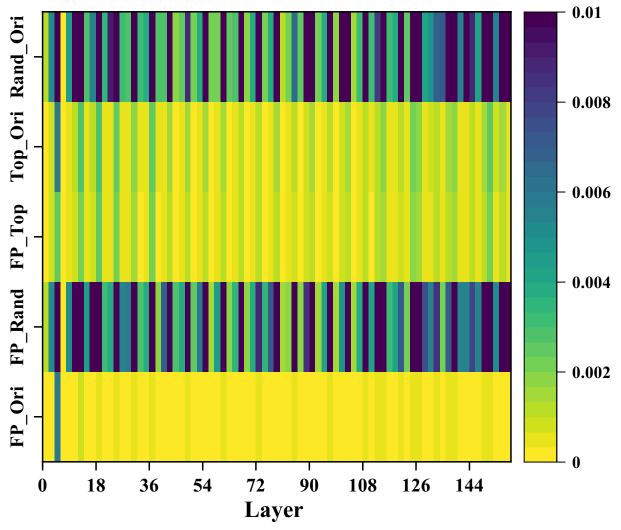

Figure 10. The analysis evaluates the reasonableness of approximating task-specific information using the most changed parameters during fine-tuning, by calculating the L1 distance of each layer's parameters in CLIP-B/32 fine-tuned on 8 visual tasks with different approximation methods. Here, *Ori* refers to the original parameters, *FP* to low-pass filtering, *Top* to maximum value approximation, and *Rand* to random sampling approximation.

Meanwhile, in this section, we discuss the amount of data required for training the router. In Tab. 13, we present the average performance across eight tasks using FREE-Merging with different sizes of training data in the fusion experiments on ViT-B/32 and ViT-B/14. It can be observed that the performance improvement becomes marginal when the training data exceeds 1%; therefore, in practical applications, we use only 1% of the data for training. If resources are even more limited, using only 0.1% of the data can still yield acceptable results.

D.7. Filtering Frequency Hyperparameter Analysis

In this section, we analyze the cutoff frequency of the Fourier high-pass filter. We perform merging using the results after fine-tuning the ViT-B/32 on eight visual classification tasks. We explore the impact of filtering different proportions of low-frequency signals on model merging.

The results are shown in the Table 14. It can be observed that filtering a certain proportion of low-frequency information significantly improves the performance of the model merging. For example, filtering 10% of the low-frequency information, despite its small proportion, leads to an impressive 12% improvement in average performance. This is similar to the analysis of low-frequency information in image processing, where low-frequency components contain most of the features in the overall information. In the context of model merging, low-frequency information represents the specialized knowledge brought by fine-tuning. While this specialized information can improve model performance on specific tasks, it severely affects the generalization ability of the model, leading to task conflicts during merging. Therefore, filtering a small proportion of low-frequency information helps minimize model specialization without significantly impacting performance, thus avoiding performance loss caused by task conflicts.

From the experimental results in Table 14, it is evident that our method is robust to the filtering ratio. Even after filtering 40% of the information, the method still demonstrates strong performance improvements, with a noticeable performance drop only after filtering 60% of the information. This strongly indicates that the method proposed in this paper is highly robust to the cutoff frequency as a hyperparameter. The robustness to hyper-parameters makes our method simple to deploy, providing a significant advantage.

D.8. Fuse to Forget

In this section, we discuss the forgetting of task-specific components caused by the merging process. In the setting explored in this paper, we aim to preserve the capabilities of multiple individual models simultaneously, so that the merged model can address multiple tasks. Therefore, in our setting, it is crucial to minimize the loss of task-specific capabilities during the fusion process. To address this issue,

Table 14. The performance of Fourier high-pass filtering at different filtering percentages.

Filtering Percentage	SUN397	Cars	RESISC45	EuroSAT	SVHN	GTSRB	MNIST	DTD	Avg.
Individual	75.3	77.7	96.1	99.7	97.5	98.7	99.7	79.4	90.5
0%	65.3	63.4	71.4	71.7	64.2	52.8	87.5	50.1	65.8
10%	66.1	65.3	77.2	90.0	84.3	81.0	98.2	58.4	77.6
20%	65.6	64.5	76.9	90.3	84.6	82.6	98.5	59.1	77.7
30%	66.2	64.5	77.2	90.1	85.4	82.3	98.5	60.0	78.1
40%	65.2	64.4	76.1	90.4	84.5	82.2	98.5	59.0	77.5
50%	64.9	64.4	76.3	88.9	82.3	81.2	98.4	58.9	76.9
60%	65.4	65.0	76.9	86.7	77.3	76.6	96.9	57.7	75.3

Table 15. Results of merging RoBERTa pre-trained models on eight datasets from GLUE benchmark.

Method	CoLA	SST2	MRPC	STSB	QQP	MNLI	QNLI	RTE	Avg
Individual	60.18	94.04	89.22	90.63	91.41	87.20	92.71	79.06	85.55
Weight Averaging	13.96	64.11	69.36	31.84	75.36	42.19	58.70	55.23	51.34
Task Arithmetic [21]	18.78	85.89	79.90	74.03	83.78	59.08	69.67	62.09	66.65
Ties-Merging [48]	20.48	84.40	81.13	58.19	85.70	64.65	74.81	42.96	64.04
Breadcrumbs [7]	**25.23**	85.21	78.19	68.45	81.07	64.40	**79.83**	63.18	68.19
PCB-Merging [12]	23.46	84.59	79.36	68.89	81.98	63.57	76.56	64.47	67.86
FR-Merging(ours)	15.25	**86.81**	**82.39**	**80.81**	**86.21**	**65.69**	78.26	**64.73**	**70.02**
RegMean [23]	36.67	90.60	75.74	62.68	83.55	70.02	82.35	58.48	70.01
EMR-Merging [20]	37.82	87.91	82.31	**72.90**	82.61	78.91	83.24	67.92	74.20
Twin-Merging [29]	52.30	93.23	**88.65**	68.64	81.99	78.80	88.40	74.37	78.29
FREE-Merging(ours)	**54.50**	**93.69**	88.46	67.04	**88.03**	**80.60**	**89.90**	**79.06**	**80.16**

we propose FR-Merging and FREE-Merging.

On the other hand, [54] suggests that such forgetting is not necessarily harmful, for example, in reducing model bias and other undesirable effects. In their setup, the goal is to actively forget certain undesirable or sensitive information through merging. Thus, the desired behavior of model fusion differs between the two settings.

However, what remains consistent is that the merging process can lead to a loss of each model's specific capabilities. As discussed earlier, this is primarily due to potential conflicts with the knowledge from other models.

D.9. Future Works

In this section, we present the directions for our future work. First, we plan to extend our current merging methods to fine-tuned models with different architectures, thereby enhancing the generalizability and applicability of our approach across a wider range of model types. In addition, we intend to conduct a deeper investigation into the differences between parameter-efficient fine-tuning (PEFT) and full fine-tuning, aiming to understand their respective characteristics and behaviors more thoroughly. Based on these insights, we will develop customized merging strategies specifically tailored to PEFT, enabling more effective integration and deployment of existing PEFT-based models.

E. Additional Experimental Results

E.1. Merging Vision Models

In this section, we comprehensively present the detailed results of model merging using ViT-L/14 as the pre-trained backbone model across all 30 diverse vision datasets introduced and described in Sec. 6.1, as shown in Table 16.

From the results, we can see that, consistent with the conclusions presented in the main text, our proposed FR-Merging method significantly outperforms all existing cost-free model merging techniques. It is able to efficiently construct a high-performance merged backbone, demonstrating strong generalization and robustness across various tasks without incurring any additional training or computational overhead. This makes FR-Merging a highly effective solution for model merging in resource-constrained settings. In addition, when we consider the dynamic addition of expert models through FREE-Merging, the method is able to maintain satisfactory performance levels while introducing only a minimal amount of extra computational cost. This flexibility allows for adaptive model expansion depending on the complexity of the task, making FREE-Merging a practical approach for real-world applications where both performance and efficiency are important considerations.

Table 16. Task-specific and average performance when merging ViT-B/16 models on **30** tasks.

Task-specific Acc	MNIST	Cifar-10	Vegetables	Food-101	Kvasir-v2	Intel-Images	Cars	EuroSAT	Weather	Cats and Dogs
Individual	99.22	97.88	100.00	87.93	94.31	94.63	85.96	99.04	98.22	99.05
Weight Averaging	27.63	42.91	83.20	68.02	25.27	82.40	7.74	24.37	61.06	91.28
Task Arithmetic [21]	30.81	59.86	91.97	73.06	31.05	89.03	9.34	31.25	74.56	93.61
Ties-Merging [48]	23.21	42.82	92.31	73.22	21.09	89.39	5.30	10.98	72.86	91.88
Breadcrumbs [7]	33.95	54.29	91.83	66.24	32.12	14.30	0.19	32.33	69.04	90.92
FR-Merging(ours)	**42.78**	**63.35**	**93.16**	**76.25**	**38.13**	**90.73**	**12.00**	**36.37**	**75.60**	**96.69**
RegMean [23]	90.71	89.65	**99.10**	76.14	71.00	93.60	16.28	74.13	86.62	98.54
AdaMerging [52]	81.22	87.54	97.97	75.23	22.76	91.02	0.42	44.60	89.13	96.91
Twin-Merging [29]	92.10	89.34	98.54	81.43	78.45	94.14	58.12	69.43	92.23	**98.85**
EMR-Merging [20]	93.40	90.23	98.34	83.54	76.54	**95.12**	54.34	76.64	94.54	98.65
FREE-Merging(ours)	**97.30**	**95.60**	99.00	**84.65**	**85.25**	94.45	**58.52**	**79.12**	**96.38**	98.62

	Dogs	Fashion	Pet	LandScape	Flowers	STL-10	CUB-200-2011	EMNIST	DTD	RESISC45
Individual	85.16	93.26	92.23	86.83	98.19	99.07	84.79	94.67	71.76	98.90
Weight Averaging	47.80	20.46	31.26	73.14	68.97	37.74	37.66	7.73	14.63	13.56
Task Arithmetic [21]	47.65	37.11	33.24	79.59	80.68	39.66	**41.86**	11.05	14.73	15.50
Ties-Merging [48]	26.03	27.05	12.84	78.27	34.33	6.17	31.28	5.61	3.71	6.79
Breadcrumbs [7]	40.91	35.29	39.16	83.80	81.94	54.73	27.58	3.25	17.81	15.20
FR-Merging(ours)	**49.86**	**42.73**	**41.26**	**85.80**	**84.49**	**59.53**	41.79	**23.80**	**22.23**	**19.49**
RegMean [23]	42.89	83.42	34.62	83.64	95.26	78.94	49.78	48.67	30.53	34.66
AdaMerging [52]	53.09	76.76	48.34	81.98	95.69	68.91	48.19	18.02	16.68	24.83
Twin-Merging [29]	72.65	80.64	68.78	85.49	96.75	74.72	60.47	57.47	41.25	50.57
EMR-Merging [20]	72.34	82.35	67.35	85.73	96.46	76.65	63.11	60.03	40.48	49.57
FREE-Merging(ours)	**75.86**	**86.43**	**72.57**	**86.42**	**97.54**	**79.01**	**65.65**	**63.64**	**45.71**	**54.25**

	MangoLeafBD	Beans	Cifar-100	GTSRB	SVHN	SUN397	KenyanFood13	Animal-10N	Garbage	Fruits-360
Individual	100.00	97.73	89.85	95.74	96.22	78.98	85.53	92.52	93.36	99.63
Weight Averaging	68.58	70.98	77.98	15.00	10.88	57.42	33.55	46.00	22.89	5.38
Task Arithmetic [21]	**87.02**	84.62	80.20	37.01	17.41	55.88	36.32	51.14	25.23	6.15
Ties-Merging [48]	76.58	67.22	78.61	40.74	10.54	52.69	19.90	19.13	3.91	1.50
Breadcrumbs [7]	79.25	85.15	75.65	38.51	22.89	47.96	26.07	17.303	28.06	1.40
FR-Merging(ours)	86.50	**87.38**	**82.03**	**44.19**	**30.08**	**56.98**	**38.90**	**49.10**	**32.01**	**12.00**
RegMean [23]	98.10	92.58	82.59	56.96	66.13	58.58	57.11	68.74	65.31	19.79
AdaMerging [52]	99.13	93.38	84.19	59.90	25.70	64.09	48.66	66.55	38.54	7.94
Twin-Merging [29]	100.0	94.24	85.57	65.87	75.49	71.25	63.81	71.67	70.82	25.56
EMR-Merging [20]	100.0	95.45	86.43	65.94	76.46	72.45	65.43	73.84	72.24	28.24
FREE-Merging(ours)	**100.0**	**96.40**	**88.65**	**68.63**	**82.23**	**76.78**	**69.45**	**78.40**	**75.84**	**37.97**

Average Acc	Individual	Weight Averaging	Task Arithmetic	Ties-Merging	Breadcrumbs	**FR-Merging(ours)**
Acc	93.02	42.52	48.89	37.53	43.57	**53.91**

Average Acc	Individual	RegMean	AdaMerging	Twin-Merging	EMR-Merging	**FREE-Merging(ours)**
Acc	93.02	68.14	60.25	75.52	76.39	**79.67**

Table 17. Results of merging $(IA)^3$ models with T0-3B as pre-trained model on eleven NLP tasks.

Method	RTE	CB	Winogrande	WiC	WSC	COPA	H-SWAG	Story Cloze	ANLI-R1	ANLI-R2	ANLI-R3	Avg. Acc
Individual	82.7	95.8	75.1	71.7	65.3	85.3	44.4	94.9	70.2	46.5	53	71.4
Traditional MTL	88.6	95.8	75.5	61.1	80.6	94.1	42.3	97.6	70.5	49.8	47.7	73.1
Fisher Merging [33]	83.3	83.3	56.7	54.2	58.3	83.1	42.2	94.1	45.9	41.0	42.2	62.2
Task Arithmetic [21]	74.1	83.3	62.8	49.1	49.3	87.5	41.5	95.3	60.8	49.4	50.0	63.9
Weight Averaging	81.2	58.3	53.8	55.2	53.5	80.9	40.1	92.5	43.3	39.2	40.2	58.0
Ties-Merging [48]	78.0	83.3	67.9	57.6	59.7	81.7	42.8	90.3	**66.9**	**51.3**	**51.1**	66.4
Breadcrumbs [7]	71.9	59.4	53.1	31.2	64.1	78.1	46.9	90.2	50.0	31.2	25.0	54.6
PCB-Merging [12]	**85.9**	83.3	61.9	57.1	63.9	82.4	42.7	91.2	64.2	47.8	45.9	66.1
FR-Merging(ours)	81.3	**83.8**	**68.3**	**57.9**	**65.2**	**88.4**	**48.6**	**93.5**	56.2	46.2	46.3	**66.9**
RegMean [23]	81.2	58.3	53.8	55.2	53.5	80.9	40.1	92.5	43.3	39.2	40.2	58.0
EMR-Merging [20]	81.8	87.5	66.6	56.1	65.3	82.4	44.7	93.6	65.7	43.8	50.8	67.1
Twin-Merging [29]	81.2	85.6	65.4	57.2	**66.3**	81.6	44.4	92.9	66.5	42.4	51.2	66.8
FREE-Merging(ours)	**83.2**	**89.3**	**69.9**	**58.4**	65.5	**84.3**	**45.2**	**94.2**	**67.9**	**45.2**	**52.4**	**68.7**

Table 18. Results of merging LoRA models with QWEN-14B as pre-trained model on three generative tasks.

Method	MMLU	TruthfulQA	BBQ	Avg.
Pre-trained	69.30	51.27	80.69	67.09
Fine-tuned	68.35	54.34	93.53	72.07
Weight Averaging	68.10	50.01	82.31	66.80
Task Arithmetic [21]	67.62	53.38	78.24	66.41
Task Arithmetic (w/ DARE) [53]	67.82	52.66	82.83	67.77
Ties-Merging [48]	68.27	50.01	**84.10**	67.46
Ties-Merging (w/ DARE) [53]	**69.32**	53.07	81.19	67.86
Breadcrumbs [7]	68.24	52.88	79.20	66.77
PCB-Merging [12]	67.23	52.48	80.37	66.69
FR-Merging(ours)	68.16	**53.39**	82.44	**68.00**
RegMean [23]	67.89	52.45	82.24	67.52
EMR-Merging [20]	67.82	55.01	90.13	70.98
Twin-Merging [29]	68.32	55.76	90.98	71.68
FREE-Merging(ours)	**68.83**	**57.39**	**92.14**	**72.78**

E.2. Merging Language Models

In this section, we present the complete and detailed experimental results of merging a diverse set of language models, each fine-tuned under different settings, across multiple widely-used benchmark datasets that cover both classification and generative tasks.

First, we present the merging results using RoBERTa as the pre-trained model, which was fine-tuned individually on the eight tasks included in the GLUE benchmark. These results are summarized in Table 15. Our proposed method achieves the best performance on nearly all of the datasets, highlighting its strong generalization capability when applied to language models. In addition, Table 17 reports the merging results across 11 language classification tasks, where T0-3B serves as the pre-trained model and IA^3 is used for fine-tuning. Our method also demonstrates a clear advantage in this setting. Finally, Table 18 provides the results for merging with QWEN-14B as the pre-trained model and LoRA as the fine-tuning technique, evaluated on three generative tasks. In all these experiments, both FR-Merging and FREE-Merging deliver consistently superior performance compared to other existing model merging methods, confirming their effectiveness across various fine-tuning strategies and model types.

Overall, our proposed methods demonstrate strong generalization capabilities when applied to language models, consistently delivering robust performance across a wide range of tasks, architectures, and fine-tuning strategies.