FroM: Frobenius Norm-Based Data-Free Adaptive Model Merging

Zijian Li[†] **Xiaocheng Feng**[†‡*] **Huixin Liu**[†]
Yichong Huang[†] **Ting Liu**[†] **Bing Qin**[†‡*]
[†]Harbin Institute of Technology [‡]Peng Cheng Laboratory
{lizijian, xcfeng, hxliu, ychuang, tliu, qinb}@ir.hit.edu.cn

Abstract

With the development of large language models, fine-tuning has emerged as an effective method to enhance performance in specific scenarios by injecting domain-specific knowledge. In this context, model merging techniques provide a solution for fusing knowledge from multiple fine-tuning models by combining their parameters. However, traditional methods often encounter task interference when merging full fine-tuning models, and this problem becomes even more evident in parameter-efficient fine-tuning scenarios. In this paper, we introduce an improvement to the RegMean method, which indirectly leverages the training data to approximate the outputs of the linear layers before and after merging. We propose an adaptive merging method called FroM, which directly measures the model parameters using the Frobenius norm, without any training data. By introducing an additional hyperparameter for control, FroM outperforms baseline methods across various fine-tuning scenarios, alleviating the task interference problem.

1 Introduction

In recent years, significant advancements have been made in the fields of artificial intelligence and natural language processing, with large language models (LLMs) emerging as a focal point of attention (Achiam et al., 2023; Guo et al., 2025). Among the various approaches, the pretraining and fine-tuning paradigm has emerged as the foundational framework for LLM technology (Zhao et al., 2023; Han et al., 2021; Parthasarathy et al., 2024). With a growing number of fine-tuned downstream models, model merging has emerged as an effective approach to integrate models trained under different hyperparameter settings or for multi-task learning (Jin et al., 2022). Moreover, model merging is commonly implemented in a data-free method, making it particularly advantageous for privacy-sensitive applications like federated learning (McMahan et al., 2017; Wang et al., 2020).

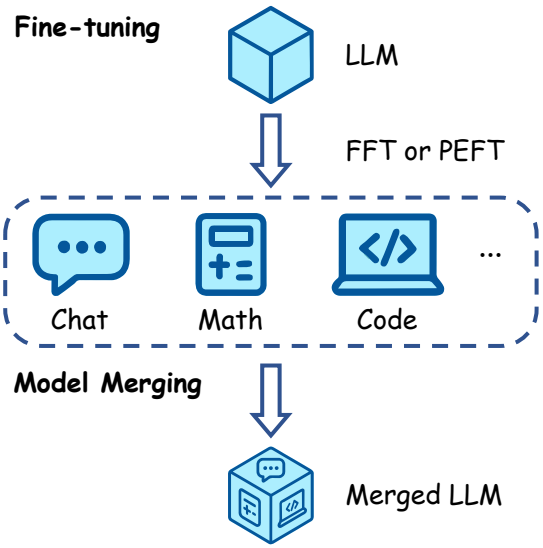

Figure 1: Illustrative diagrams of fine-tuning and model merging.

Most existing model merging methods are based on linear operations such as weighted averaging (Wortsman et al., 2022) and task arithmetic (Ilharco et al., 2022). Weighted averaging combines parameters from multiple fine-tuned models, while task arithmetic enables multi-task learning and forgetting by adding and subtracting task vectors derived from parameter differences. Nevertheless, these methods encounter a fundamental challenge: simple linear combinations of parameters often lead to task interference (Yadav et al., 2024). This issue is especially evident in parameter-efficient fine-tuning (PEFT) scenarios (Ding et al., 2023), such as Low-Rank Adaptation (LoRA) (Hu et al., 2021). Compared to full fine-tuning (FFT), the sparse structure of low-rank matrices in these methods further aggravates task interference (Tang et al., 2023; Stoica et al., 2024). Figure 1 presents a

*Corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 4674–4687

schematic illustration of the fine-tuning and model merging processes.

To address this challenge, we take inspiration from the nonlinear weighted optimization method of RegMean (Jin et al., 2022) and introduce a novel method, **Fro**benius Norm-Based Data-Free Adaptive Model **M**erging (FroM). Our method improves upon the RegMean method, which derives a closed-form solution by approximating the linear layer outputs of the models using the inner product matrix of the training data. However, RegMean still encounters challenges due to its reliance on access to training data, which may not always be available. To address this limitation, FroM adopts a data-free approach that directly measures model parameters using the Frobenius norm, and employs adaptive coefficients to adjust the discrepancies between task vectors. Our approach reduces the reliance on training data, while outperforming baselines in both full fine-tuning and LoRA scenarios, effectively mitigating the task interference issue.

Our main contributions are as follows:

1. We propose an adaptive model merging method, FroM, which utilizes the Frobenius norm to measure the differences between models to be merged, without any training data.

2. Compared to baseline methods, our approach is broadly applicable to FFT and PEFT models, such as LoRA, and achieves optimal or suboptimal results in various scenarios.

2 Related Work

Fine-tuning of LLMs. Fine-tuning is a method for optimizing pre-trained LLMs for specific tasks. Through fine-tuning, LLMs not only excel in general tasks but can also meet the requirements of particular application scenarios (Wei et al., 2021; Sanh et al., 2021; Chung et al., 2024). From the perspective of the amount of parameters updated, fine-tuning can be categorized into two approaches: FFT and PEFT. FFT updates all parameters of the pre-trained model, which often yields excellent performance on specific tasks. However, this approach is both time-consuming and computationally intensive. As models become larger and more complex, conducting FFT has become more challenging, particularly in environments with limited computational resources (Lv et al., 2023). Alternatively, PEFT focuses on updating a small number of parameters to reduce computational costs and storage requirements, making it more widely used in resource-constrained settings or customized scenarios (Xu et al., 2023b; Han et al., 2024). One of the most widely used methods within PEFT is LoRA (Hu et al., 2021), which significantly reduces computational and storage costs by fine-tuning low-rank matrices, achieving highly effective fine-tuning with a limited number of parameters.

Model Merging. (Lu et al., 2024) introduce several collaboration strategies for LLMs, highlighting model merging as a technique that prevents models from getting trapped in local optima and improves multi-task collaboration ability. Traditional methods typically rely on weighted averaging of model parameters. (Wortsman et al., 2022) introduce Model Soups, which apply both a weighted averaging method and a greedy merging strategy. (Matena and Raffel, 2022) adopt a different approach by utilizing the Fisher Information Matrix for parameter-level model merging, applying a weighted merging strategy to enhance model integration. Focusing on the outputs of linear layers, (Jin et al., 2022) propose RegMean, a method that indirectly utilizes the training data through the inner product matrices to derive a closed-form solution for merging model parameters. Meanwhile, (Ilharco et al., 2022) present a novel paradigm called task arithmetic, in which task vectors capture the ability shift of different fine-tuned models compared to the pre-trained model across various tasks. This allows for operations like forgetting and learning through arithmetic manipulation of the task vectors.

However, the task vectors of different models may inevitably lead to some degree of task interference. Prior research has demonstrated that weight disentanglement is crucial to the success of task arithmetic (Ortiz-Jimenez et al., 2023). To address this, (Yadav et al., 2024) introduce TIES-Merging, a strategy that minimizes redundant parameters and selects signs to reduce interference during model merging. In addition, the DARE method proposed by (Yu et al., 2024) involves the random dropping of incremental parameters and rescaling operations. By integrating with other model merging algorithms, it enhances the performance of the model after fusion. According to the research by (Tang et al., 2023; Stoica et al., 2024), existing merging methods are more challenging to apply to LoRA fine-tuned models compared to FFT models. Our method builds upon RegMean and introduces improvements that eliminate the need for

obtaining the inner product matrix of training data. It can also be solved using a closed-form solution, supporting the integration of either FFT models or LoRA adapters.

3 Method

3.1 Motivation

The RegMean method naturally aims to make the output of the merged model close to the outputs of the original models. It cleverly uses the inner product matrix derived from the training data through a closed-form solution, while avoiding data privacy issues that may arise from directly using the training data. However, this brings about additional requirements for model release, as open-source models typically do not provide the inner product matrix. This limitation significantly impacts the applicability of the RegMean method.

Therefore, we propose a different approach: instead of measuring the output discrepancies between the merged model and the original models, why not directly measure the model weights themselves? However, directly measuring model weights is equivalent to the weighted averaging method, which has limited effectiveness. We draw inspiration from research on model pruning, where compression is achieved by removing connections with small weights (Han et al., 2015). This aligns with our intuition that larger task vectors generally contain more information. We adopt a holistic, layer-wise metric based on the squared Frobenius norm, which measures the overall magnitude of weight matrices and serves as an effective indicator of model differences. For this reason, we use the Frobenius norm of the task vector to quantify model differences and raise it to the power of k to serve as a weighting coefficient, with k as a hyperparameter controlling this balance.

3.2 Merging of FFT Models

We use the objective function in Equation 1 to measure the performance difference between the task vectors before and after merging.

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \|\boldsymbol{\theta} - \boldsymbol{\theta}_i\|_F^2 \tag{1}$$

where n denotes the number of fine-tuned models, $\boldsymbol{\theta}_i$ represents the task vector of the i-th model, $\boldsymbol{\theta}$ is the task vector after merging, and k is a hyperparameter. When $k = 0$, the objective function

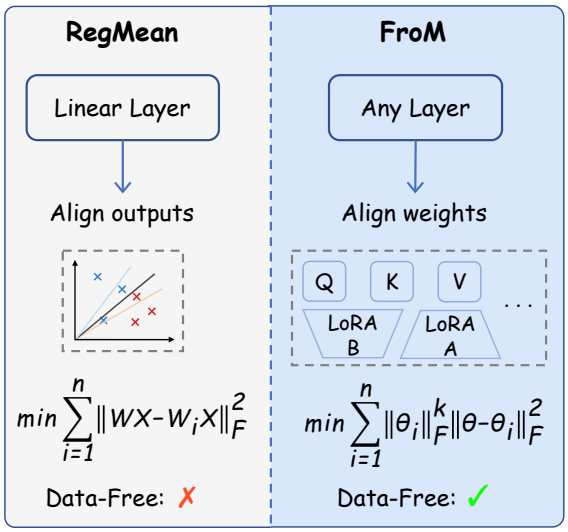

Figure 2: Comparison between the RegMean and FroM Methods, where X represents the input to the linear layer.

of FroM serves as an upper bound for that of RegMean: $\sum_{i=1}^{n} \|\boldsymbol{W}\boldsymbol{X} - \boldsymbol{W}_i\boldsymbol{X}\|_F^2 \leq \sum_{i=1}^{n} \|\boldsymbol{W} - \boldsymbol{W}_i\|_F^2 \cdot \|\boldsymbol{X}\|_F^2 = \sum_{i=1}^{n} \|\boldsymbol{\theta} - \boldsymbol{\theta}_i\|_F^2 \cdot \|\boldsymbol{X}\|_F^2$, where $\|\boldsymbol{X}\|_F^2$ acts as a scaling factor. Moreover, by setting $k > 0$, FroM enables more flexible control in determining an appropriate trade-off point. The comparison between the RegMean and FroM Methods is illustrated in Figure 2.

Since the above function is convex, a closed-form solution (i.e., the point where the gradient is zero) can be derived. The objective function can be restructured, resulting in:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}\left((\boldsymbol{\theta} - \boldsymbol{\theta}_i)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}_i)\right) \tag{2}$$

$$= \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}\left(\boldsymbol{\theta}^\top \boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \boldsymbol{\theta}_i + \boldsymbol{\theta}_i^\top \boldsymbol{\theta}_i\right) \tag{3}$$

The last term in the parentheses is a constant; therefore, the derivative with respect to $\boldsymbol{\theta}$ is:

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \boldsymbol{\theta} - 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \boldsymbol{\theta}_i \tag{4}$$

By setting $\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0$, the closed-form solution is derived, as shown in Equation 5. In fact, we observe that as $k \to 0$, the final result corresponds to the simple averaging method, whereas as $k \to +\infty$, the result converges to the method that selects the weight with the largest norm. Therefore, this

approach can be viewed as a balance between these two methods.

$$\boldsymbol{\theta}^* = \frac{\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \boldsymbol{\theta}_i}{\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k} \tag{5}$$

3.3 Merging of LoRA

The task arithmetic-based approach poses challenges in integrating LoRA (Tang et al., 2023; Stoica et al., 2024), making it difficult both to merge and to control the rank of the merged model. To address this issue, we modify the objective function based on Equation 1, incorporating LoRA-type structures while measuring weight differences using the Frobenius norm. The revised objective function is given in Equation 6, where θ_i refers to the task vector in its original form before being flattened into a one-dimensional vector. When merging parameters layer-wise across different models, we employ LoRA-type structures, specifically utilizing the $\mathbf{A}$ and $\mathbf{B}$ matrices as components optimized in a manner similar to the LoRA mechanism. Moreover, by explicitly specifying the rank of $\mathbf{A}$ and $\mathbf{B}$ (i.e., the row count of $\mathbf{A}$ and the column count of $\mathbf{B}$), we enable dynamic rank adjustment. This approach not only increases flexibility in model merging but also facilitates a certain degree of compression during model storage.

$$\mathcal{L}(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \|\mathbf{B}\mathbf{A} - \boldsymbol{\theta}_i\|_F^2 \tag{6}$$

Since Equation 6 is a non-convex function, it may have multiple local minima. Therefore, inspired by the Alternating Direction Method of Multipliers in convex optimization theory and analogous to LoRM (Salami et al., 2024), we propose an alternating optimization algorithm. Specifically, by initializing $\mathbf{A}$ from a normal distribution with mean 0 and variance σ^2, and treating either $\mathbf{A}$ or $\mathbf{B}$ as an invariant, we iteratively update $\mathbf{B}$ and $\mathbf{A}$ using Equation 7 and Equation 8, respectively. In these equations, † denotes the Moore-Penrose inverse (see Appendix A for the proof). This procedure for approximating a local minimum of the objective function is shown in Algorithm 1. Due to the potential computational errors in the calculation of the MP inverse, an early stopping mechanism is implemented, which halts the iteration when the loss begins to increase.

$$\mathbf{B} = \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \boldsymbol{\theta}_i \mathbf{A}^\top\right) \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{A}\mathbf{A}^\top\right)^\dagger \tag{7}$$

$$\mathbf{A} = \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}^\top\mathbf{B}\right)^\dagger \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}^\top \boldsymbol{\theta}_i\right) \tag{8}$$

3.4 Complexity

We next analyze the computational complexity of the two merging methods discussed above. We let n denote the number of models to be merged, and d represent the number of parameters in the FFT models. Therefore, the complexity of merging FFT models is $O(nd)$. For the merging of LoRA (i.e., Algorithm 1), we simplify the analysis by assuming that LoRA is applied to every layer (in practice, it is typically applied to only a subset of layers). We let L denote the number of layers in the model and T the number of iterations, assuming that each layer's weight matrix has dimensions (d_1, d_2), with the corresponding LoRA matrices B and A are of sizes (d_1, r) and (r, d_2), respectively. The first term in Equation 7 involves matrix multiplication with complexity $O(d_2^2 r)$. The second term involves both matrix multiplication and matrix inversion, with respective complexities $O(d_2 r^2)$ and $O(r^3)$. Thus, the overall complexity becomes: $O(d_2^2 r + d_2 r^2 + r^3) = O(d_2^2 r)$, since $r \ll min(d_1, d_2)$. Similarly, the complexity of Equation 8 is $O(d_1^2 r)$. Therefore, the overall complexity of the algorithm is: $O(nLTr(d_1^2 + d_2^2))$.

4 Experimental Setup

Merging of FFT Models. We test two different model architectures, each based on the same foundational model, across three distinct FFT tasks: Instruction-Following, Mathematical Reasoning, and Code-Generating. First, we adopt the experimental setup of (Yu et al., 2024), utilizing models based on Llama-2-13B (Touvron et al., 2023): WizardLM-13B (Xu et al., 2023a), WizardMath-13B (Luo et al., 2023), and llama-2-13b-code-alpaca[1]. Second, we utilize models based on Qwen2.5-7B[2] (Yang et al., 2024): Qwen2.5-

[1]https://huggingface.co/layoric/llama-2-13b-code-alpaca
[2]https://huggingface.co/Qwen/Qwen2.5-7B

Algorithm 1 Alternating Optimization-Based Model Merging Algorithm Utilizing LoRA-Type Structures

Initialize matrix $\mathbf{A} \sim \mathcal{N}(0, \sigma^2)$
Initialize early stopping flag `early_stopping`

Set iteration count `iters`
`min_loss` $\leftarrow \infty$ {Initialize minimum loss}
for $i = 0$ to `iters` $- 1$ **do**
  Compute matrix $\mathbf{B}$ using Equation 7
  Compute matrix $\mathbf{A}$ using Equation 8
  if `early_stopping` **then**
    Compute `loss` using Equation 6
    if `loss` < `min_loss` **then**
      `min_loss` $\leftarrow$ `loss`
    else
      `break`
    end if
  end if
end for
Save matrices $\mathbf{A}$ and $\mathbf{B}$

7B-Instruct[3], Qwen2.5-Math-7B-Instruct[4], and Qwen2.5-Coder-7B-Instruct[5]. For testing, we employ the Task Arithmetic (Ilharco et al., 2022), TIES-Merging (Yadav et al., 2024), and our FroM methods, additionally incorporating the DARE method (Yu et al., 2024) in combination with TIES-Merging and FroM for comparative evaluation. We do not include the RegMean method in our comparison, as we are unable to obtain the inner product matrices of all model training datasets, which highlights a significant practical limitation of RegMean. For both evaluation settings involving different base models, we set the linear weighting coefficient α to 1.0 (see Appendix B.2 for details), and use $k = 1.0$ in our FroM method. For the choice of the hyperparameter k in practical applications, refer to Section 5.3.

Regarding benchmark selection, we follow the setup of (Yu et al., 2024), utilizing AlpacaEval (Li et al., 2023) to assess the models' instruction-following abilities, GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) to evaluate their mathematical reasoning skills, and HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) for testing code generation capabilities. Further details can be found in Appendix B.

Merging of LoRA. We fine-tune a binary classification head on top of the Meta-Llama-3-8B[6] model. First, we initialize the classifier by training it on the MNLI dataset from the GLUE benchmark (Wang et al., 2019). Then, we fine-tune the model using the binary text entailment datasets from the GLUE benchmark, specifically QNLI, RTE, and WNLI, with the LoRA approach. For each task, we train the model for 10 epochs and select the best checkpoint for evaluation. We not only test the merging of LoRA across three tasks but also examine the merging of the three optimal checkpoints for the same task. We apply the Task Arithmetic, RegMean, TIES-Merging, DARE method (which combines the TIES-Merging approach), KnOTS method (which integrates the first three methods) (Stoica et al., 2024), and FroM method (with $k = 0.9$). The linear weighting coefficient is set to $\alpha = 0.7$, determined via a linear search. Further details, including the search results and relevant settings, are provided in Appendix B.

5 Main Results

5.1 Merging of FFT Models

The merging results of the WizardLM-13B, WizardMath-13B, and llama-2-13b-code-alpaca models based on Llama-2-13b are shown in Table 1. As observed, for the merging of the first two models, our FroM method achieves better average performance, outperforming the previous optimal baseline by 0.73%. Similarly, when merging WizardLM-13B and llama-2-13b-code-alpaca, FroM also achieves the highest average score. For the merging of the latter two models, however, the Task Arithmetic and TIES-Merging methods demonstrate better performance. In the case of merging three models, the Task Arithmetic method performs best on the AlpacaEval benchmark, but shows moderate performance across the other four datasets. For tasks related to Mathematical Reasoning and Code-Generating, both our FroM method and its integration with DARE exhibit strong performance, with the latter achieving the best average performance, surpassing the previous optimal baseline by 1.68%. Therefore, although the FroM method does not consistently outperform the baseline across every benchmark, a careful se-

[3]https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
[4]https://huggingface.co/Qwen/Qwen2.5-Math-7B-Instruct
[5]https://huggingface.co/Qwen/Qwen2.5-Coder-7B-Instruct
[6]https://huggingface.co/meta-llama/Meta-Llama-3-8B

Models	Merging Methods	Instruction-Following	Mathematical Reasoning		Code-Generating		Avg.
		AlpacaEval	GSM8K	MATH	HumanEval	MBPP	
LM	/	51.09	45.11	5.92	32.93	31.20	33.25
Math	/	/	59.97	11.60	/	/	35.78
Code	/	/	/	/	24.39	28.00	26.20
LM & Math	Task Arithmetic	50.55	41.62	6.56	/	/	32.91
	TIES-Merging	50.42	51.55	7.22	/	/	36.40
	DARE+TIES-Merging	53.03	52.69	**8.32**	/	/	38.01
	FroM	50.82	**57.39**	8.00	/	/	**38.74**
	DARE+FroM	**53.65**	53.75	7.58	/	/	38.33
LM & Code	Task Arithmetic	**53.23**	/	/	32.93	30.60	38.92
	TIES-Merging	46.33	/	/	0.00	0.00	15.44
	DARE+TIES-Merging	47.67	/	/	0.00	0.00	15.89
	FroM	52.33	/	/	34.76	**33.40**	**40.16**
	DARE+FroM	50.40	/	/	**35.37**	32.60	39.45
Math & Code	Task Arithmetic	/	**58.45**	**12.32**	5.49	7.00	20.82
	TIES-Merging	/	58.00	12.28	**9.15**	**19.00**	**24.61**
	DARE+TIES-Merging	/	57.32	12.10	6.10	18.40	23.48
	FroM	/	58.15	11.98	4.88	17.40	23.10
	DARE+FroM	/	58.07	12.30	8.54	16.80	23.93
LM & Math & Code	Task Arithmetic	**52.84**	42.00	6.20	15.85	18.20	27.02
	TIES-Merging	46.24	47.08	5.32	0.00	0.00	19.73
	DARE+TIES-Merging	48.30	55.50	**9.18**	25.00	27.00	33.00
	FroM	49.65	**56.48**	8.20	25.61	31.40	34.27
	DARE+FroM	49.12	55.34	8.06	**29.09**	**31.80**	**34.68**

Table 1: Merging results of WizardLM-13B, WizardMath-13B, and llama-2-13b-code-alpaca models based on Llama-2-13B. Bold and underlined text indicate the optimal and suboptimal results, respectively.

Models	Merging Methods	Instruction-Following	Mathematical Reasoning		Code-Generating		Avg.
		AlpacaEval	GSM8K	MATH	HumanEval	MBPP	
LM	/	16.24	79.08	14.40	79.88	65.40	51.00
Math	/	/	75.74	1.12	/	/	38.43
Code	/	/	/	/	30.49	39.00	34.74
LM & Math	Task Arithmetic	**2.25**	**18.95**	**6.04**	/	/	**9.08**
	TIES-Merging	0.00	0.68	0.00	/	/	0.23
	DARE+TIES-Merging	0.00	0.15	0.00	/	/	0.05
	FroM	1.95	17.51	4.08	/	/	7.85
	DARE+FroM	0.00	0.45	0.00	/	/	0.15
LM & Code	Task Arithmetic	**42.76**	/	/	21.34	29.60	31.23
	TIES-Merging	0.00	/	/	0.00	0.00	0.00
	DARE+TIES-Merging	0.00	/	/	0.00	0.00	0.00
	FroM	36.98	/	/	**30.49**	**42.00**	**36.49**
	DARE+FroM	0.00	/	/	0.00	0.00	0.00
Math & Code	Task Arithmetic	/	0.38	0.00	0.00	0.00	0.09
	TIES-Merging	/	1.36	0.00	0.00	0.00	0.34
	DARE+TIES-Merging	/	0.38	0.00	0.00	0.00	0.09
	FroM	/	**17.36**	0.00	**1.22**	**0.80**	**4.85**
	DARE+FroM	/	0.38	0.00	0.00	0.00	0.09
LM & Math & Code	Task Arithmetic	0.00	0.45	0.00	0.00	0.00	0.09
	TIES-Merging	0.00	0.08	0.00	0.00	0.00	0.02
	DARE+TIES-Merging	0.00	0.23	0.00	0.00	0.00	0.05
	FroM	**1.22**	**20.55**	**0.84**	**2.44**	**2.80**	**5.57**
	DARE+FroM	0.00	0.61	0.00	0.00	0.00	0.12

Table 2: Merging results of Qwen2.5-7B-Instruct, Qwen2.5-Math-7B-Instruct, and Qwen2.5-Coder-7B-Instruct models based on Qwen2.5-7B. Bold and underlined text indicate the optimal and suboptimal results, respectively.

lection of the hyperparameter k enables a well-balanced trade-off, ultimately leading to superior performance compared to other baseline methods.

For the base model Qwen2.5-7B, we present the results of merging the models Qwen2.5-7B-Instruct, Qwen2.5-Math-7B-Instruct, and Qwen2.5-Coder-7B-Instruct, as shown in Table 2. Unexpectedly, the performance of the merged models deteriorates significantly, except when merging Qwen2.5-7B-Instruct and Qwen2.5-Coder-7B-Instruct using the Task Arithmetic or FroM methods, which yield comparatively better results. When the other two models are integrated with Qwen2.5-Math-7B-Instruct, a substantial decline in performance is observed. Based on the specific test outputs, we observe that the majority of the model's responses are either incoherent or irrelevant to the given task. This suggests a significant degree of task interference between Qwen2.5-Math-7B-Instruct and the other two models. After extensive fine-tuning on different datasets, the linear mode connectivity (Ainsworth et al., 2022; Zhou et al., 2023b) among these models tends to be disrupted, leading to the failure of linear weighting methods. This finding indicates that not all fine-tuned models derived from the same base model are equally suitable for model merging. Furthermore, the merging results using the TIES-Merging and DARE methods are close to zero (with outputs consisting entirely of garbled text), indicating that these methods aggravate task interference. In contrast, our FroM method demonstrates greater robustness when merging both two and three models.

Since the training data of these FFT models are not fully released, it is impossible to directly compare the FroM and RegMean methods in Table 1 and Table 2. That said, we fully agree that including RegMean as a baseline is crucial for a more comprehensive comparison. To this end, we conduct additional experiments with RegMean, with detailed results presented in Appendix C.

5.2 Merging of LoRA

For the LoRA merging experiments, we train a shared classification head on the MNLI dataset. The performance of this classification head on other datasets is presented in the first row of Table 3. The accuracy across different tasks exceeds 50%, demonstrating a reasonable degree of generalization, considering that for binary classification tasks, the probability of randomly selecting the correct answer is 50%. We further fine-tune on the QNLI, RTE, and WNLI datasets, selecting the best-performing checkpoints for subsequent experiments.

The testing results for different LoRA merging methods are summarized in Table 3. It is evident that the application of the TIES-Merging and DARE methods results in a decline in model accuracy, indicating that these approaches may not be universally effective across all downstream tasks and may perform particularly poorly on simpler ones. On the other hand, our FroM method continues to demonstrate effectiveness, with its average performance surpassing other baselines. Notably, for the QNLI task, our method even exceeds the baseline accuracy, highlighting that our approach enhances the generalization capability of the merged model resulting from multi-task learning.

Merging Methods	QNLI	RTE	WNLI	Avg.
Baseline				
/	53.74	59.21	54.93	55.96
Fine-tuning				
QNLI	94.05	56.32	43.66	64.68
RTE	54.75	71.49	64.79	63.68
WNLI	54.26	62.82	54.69	57.26
3-Model Merging				
Task Arithmetic	90.10	**67.51**	52.11	69.91
RegMean	69.84	66.34	53.52	63.23
TIES-Merging	65.50	65.70	**61.97**	64.39
DARE+TIES-Merging	54.26	60.65	56.34	57.08
KnOTS+Task Arithmetic	94.44	66.43	50.70	70.52
KnOTS+TIES-Merging	93.65	64.98	52.11	70.25
KnOTS+DARE+TIES-Merging	59.07	63.54	60.56	61.06
FroM	**94.51**	64.26	53.52	**70.76**

Table 3: Test results of three models fine-tuned using LoRA on QNLI, RTE, and WNLI datasets with different merging methods. Bold text indicates the optimal results.

We also test the merging of the three best checkpoints from the same dataset. As shown in Figure 4, our FroM method achieves the best performance, surpassing the accuracy of the original fine-tuning across all three datasets. For comparison, the original KnOTS method performs comparably to our approach, while Task Arithmetic yields significantly inferior results. The other methods demonstrate relatively moderate performance. This further demonstrates the effectiveness of our method in merging checkpoints for the same task.

5.3 Ablation Study

For the WizardLM-13B, WizardMath-13B, and llama-2-13b-code-alpaca models, we test the effect

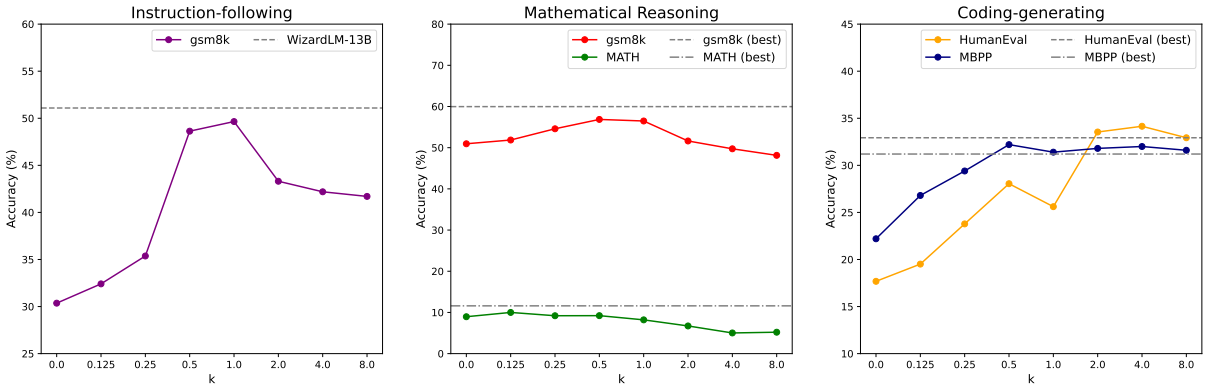

Figure 3: Results of merging WizardLM-13B, WizardMath-13B, and LLaMA-2-13B-Code-Alpaca models with different values of k. The optimal results before merging are represented by the gray dashed line.

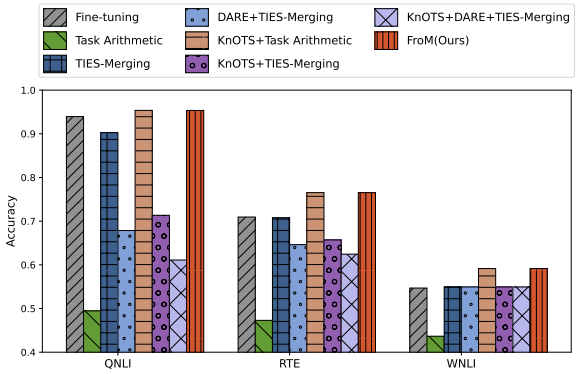

Figure 4: Accuracy comparison after merging the three optimal LoRA checkpoints on QNLI, RTE, and WNLI datasets.

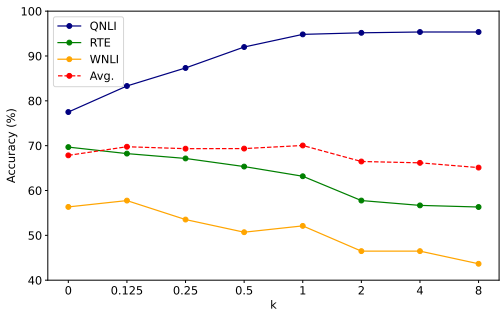

Figure 5: Results of merging the three LoRAs. The average accuracy is depicted by a dashed line.

of varying the value of k in Equation 1. Specifically, we select $k = 0$ and integer powers of 2 ranging from 2^{-3} to 2^3, and plot the results under these different conditions. As shown in Figure 3, the performance is generally poor when $k \to 0$ or $k \to +\infty$. The best results occur when k is set to 0.5 or 1.0, with accuracy on all benchmarks closely matching the optimal performance of the fine-tuned models.

We also test the effect of varying the value of k on LoRA fusion, as shown in Figure 5. The optimal average accuracy is achieved when k is set to 1.0. As the value of k increases, the performance of the fused model improves on the QNLI dataset but decreases on the RTE and WNLI datasets. This change can be attributed to the larger size of the QNLI dataset, which results in greater influence of its fine-tuned model parameters during merging.

6 Conclusion

In this paper, we propose a novel merging approach called FroM, which utilizes the Frobenius norm to quantify weight discrepancies between models without any training data. This method adjusts adaptively through a hyperparameter, and the merged parameters are derived using a closed-form solution. For the experimental part, we first test our method on three models based on Llama-2-13B, and the results show that FroM outperforms existing baseline methods. Next, while baseline methods struggle to effectively integrate three fine-tuned models based on Qwen2.5-7B, our approach outperforms the baseline methods, demonstrating its effectiveness and robustness. Finally, when merging LoRA adapters, FroM also shows superior performance over other methods. Overall, the proposed FroM method demonstrates outstanding performance in both FFT and LoRA scenarios, effectively alleviating the task interference issue in model merging and showcasing strong applicability. We hope this work inspires future research in the model merging field, helps address existing task interference issues, and provides a new perspective for tackling these challenges.

Limitations

The limitations of our FroM method can be summarized as follows: (1) the necessity for a more comprehensive theoretical analysis of the algorithm, and (2) the crucial selection of the hyperparameter k. Future research directions with significant potential include providing an interpretable metric for task interference. By analyzing the upper bound of the model's performance, it may be possible to assess and determine whether the model merging process is feasible.

Acknowledgments

Xiaocheng Feng and Bing Qin are the co-corresponding authors of this work. We thank the anonymous reviewers for their insightful comments. This work was supported by the National Natural Science Foundation of China (NSFC) (grant 62276078, U22B2059), the Key R&D Program of Heilongjiang via grant 2022ZX01A32, and the Fundamental Research Funds for the Central Universities (XNJKKGYDJ2024013).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2022. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Röttger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. 2023. Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. *arXiv preprint arXiv:2309.07875*.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open*, 2:225–250.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2022. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju

Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-CoT](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf).

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Jinliang Lu, Ziliang Pang, Min Xiao, Yaochen Zhu, Rui Xia, and Jiajun Zhang. 2024. Merge, ensemble, and cooperate! a survey on collaborative strategies in the era of large language models. *arXiv preprint arXiv:2407.06089*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Kai Lv, Yuqing Yang, Tengxiao Liu, Qinghui Gao, Qipeng Guo, and Xipeng Qiu. 2023. Full parameter fine-tuning for large language models with limited resources. *arXiv preprint arXiv:2306.09782*.

Kaifeng Lyu, Haoyu Zhao, Xinran Gu, Dingli Yu, Anirudh Goyal, and Sanjeev Arora. 2024. Keeping LLMs aligned after fine-tuning: The crucial role of prompt templates. *arXiv preprint arXiv:2402.18540*.

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

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. *Advances in Neural Information Processing Systems*, 36:66727–66754.

Venkatesh Balavadhani Parthasarathy, Ahtsham Zafar, Aafaq Khan, and Arsalan Shahid. 2024. The ultimate guide to fine-tuning llms from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *arXiv preprint arXiv:2408.13296*.

Riccardo Salami, Pietro Buzzega, Matteo Mosconi, Jacopo Bonato, Luigi Sabetta, and Simone Calderara. 2024. Closed-form merging of parameter-efficient modules for federated continual learning. *arXiv preprint arXiv:2410.17961*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. 2024. Model merging with svd to tie the knots. *arXiv preprint arXiv:2410.19735*.

Anke Tang, Li Shen, Yong Luo, Yibing Zhan, Han Hu, Bo Du, Yixin Chen, and Dacheng Tao. 2023. Parameter efficient multi-task model fusion with partial linearization. *arXiv preprint arXiv:2310.04742*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. 2020. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

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. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023a. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. 2023b. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arXiv preprint arXiv:2312.12148*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2024. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023a. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Zhanpeng Zhou, Yongyi Yang, Xiaojiang Yang, Junchi Yan, and Wei Hu. 2023b. Going beyond linear mode connectivity: The layerwise linear feature connectivity. *Advances in neural information processing systems*, 36:60853–60877.

Appendix

A Proof of the expression

The objective function given in Equation 6 can be reformulated as follows:

$$\begin{aligned}
\mathcal{L}(\mathbf{A}, \mathbf{B}) &= \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \|\mathbf{B}\mathbf{A} - \boldsymbol{\theta}_i\|_F^2 \\
&= \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}\left((\mathbf{B}\mathbf{A} - \boldsymbol{\theta}_i)^\top (\mathbf{B}\mathbf{A} - \boldsymbol{\theta}_i)\right) \\
&= \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}\left(\mathbf{A}^\top \mathbf{B}^\top \mathbf{B}\mathbf{A} - \boldsymbol{\theta}_i^\top \mathbf{B}\mathbf{A} - \mathbf{A}^\top \mathbf{B}^\top \boldsymbol{\theta}_i + \boldsymbol{\theta}_i^\top \boldsymbol{\theta}_i\right) \\
&= \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}(\mathbf{A}^\top \mathbf{B}^\top \mathbf{B}\mathbf{A}) - 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}(\mathbf{A}^\top \mathbf{B}^\top \boldsymbol{\theta}_i) + \sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \operatorname{Tr}\left(\boldsymbol{\theta}_i^\top \boldsymbol{\theta}_i\right)
\end{aligned}$$

Treating matrix $\mathbf{A}$ as a constant, we differentiate matrix $\mathbf{B}$ with respect to it, noting that the final term is a constant, which yields:

$$\frac{\partial \mathcal{L}(\mathbf{A}, \mathbf{B})}{\partial \mathbf{B}} = 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}\mathbf{A}\mathbf{A}^\top - 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \boldsymbol{\theta}_i \mathbf{A}^\top$$

Let $\frac{\partial \mathcal{L}(\mathbf{A},\mathbf{B})}{\partial \mathbf{B}} = 0$, then:

$$\mathbf{B} = \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \boldsymbol{\theta}_i \mathbf{A}^\top\right) \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{A}\mathbf{A}^\top\right)^\dagger$$

Similarly, treating matrix $\mathbf{B}$ as a constant, we differentiate with respect to matrix $\mathbf{A}$, yielding:

$$\frac{\partial \mathcal{L}(\mathbf{A}, \mathbf{B})}{\partial \mathbf{A}} = 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}^\top \mathbf{B}\mathbf{A} - 2\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}^\top \boldsymbol{\theta}_i$$

Let $\frac{\partial \mathcal{L}(\mathbf{A},\mathbf{B})}{\partial \mathbf{A}} = 0$, then:

$$\mathbf{A} = \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}^\top \mathbf{B}\right)^\dagger \left(\sum_{i=1}^{n} \|\boldsymbol{\theta}_i\|_F^k \mathbf{B}^\top \boldsymbol{\theta}_i\right)$$

B Experimental Details

B.1 Fine-tuning Settings

For the integration of LoRA adapters, the hyperparameters are set as follows: the ranks of matrices A and B are fixed at 16, the batch size is set to 32, and the learning rate is set to $1e-5$. We first train a classification head using the MNLI dataset. Since MNLI is a three-class task, we modify it to a binary classification output by combining the `neutral` and `contradiction` categories into a single `not_entailment` class. For each task within the QNLI, RTE, and WNLI datasets, we train the models for 10 epochs and select the checkpoint with the highest accuracy for subsequent LoRA adapter integration.

B.2 Test Details

For the integration of FFT models, we perform a linear search over the linear weighting coefficient α used to merge the three Llama-2-13B-based models. The results are presented in Table 4. Based on these results, we select $\alpha = 1.0$ as the final linear weighting coefficient to achieve a more balanced outcome. For the Qwen2.5-7B-based models, due to their poor merging performance, we also adopt $\alpha = 1.0$ without extensive tuning.

Regarding the evaluation, we conducted tests using AlpacaEval, GSM8K, MATH, HumanEval, and MBPP datasets. In the case of AlpacaEval, the model's performance is evaluated using the gpt-3.5-turbo model. Additionally, the final success rate metric selected is `length_controlled_winrate`, rather than `win_rate`. When merging models from the Qwen-2.5 series, we minimized the use of line breaks in the prompts. Excessive line breaks negatively impact the model's responses, leading to the generation of repetitive and meaningless text.

When merging LoRA adapters, we also employ a linear search strategy to determine the optimal linear weighting coefficient α and the hyperparameter k used in our FroM method. Based on this search, we select $\alpha = 0.7$ and $k = 0.9$. Detailed results are presented in Table 5 and 6.

C Additional Comparison with RegMean

We evaluate scenarios involving the merging of three and four models, in order to better demonstrate the generalization capability of our FroM method. Specifically, we fine-tune the Meta-Llama-3-8B base model on four tasks using supervised fine-tuning: Instruction Following, Mathematical Reasoning, Code Generation, and Safety. The corresponding datasets are Alpaca-cleaned (Taori et al., 2023), NuminaMath-CoT (LI et al., 2024), Code Alpaca (Chaudhary, 2023), and Saferpaca (Bianchi et al., 2023). For evaluation across these four dimensions, we use IFEval (Zhou et al., 2023a), GSM8K, HumanEval, and DirectHarm4 (Lyu et al., 2024), where the metric is the proportion of safe responses. We merge three and four models to better evaluate the generalization ability of these methods. The results are shown in Table 7. Since RegMean leverages a large amount of training data, FroM does not outperform RegMean but still achieves better results than the other methods.

The results of the linear search for the linear weighting coefficient and the hyperparameter used in FroM within the range $[0.5, 1.0]$ are presented in Table 8 and Table 9. The final choices are $\alpha = 0.6$ and $k = 0.5$.

α	AlpacaEval	GSM8K	MATH	HumanEval	MBPP	Avg.
0.5	49.76	61.79	**10.32**	6.71	4.60	26.64
0.6	53.88	**63.53**	9.60	8.54	3.00	27.71
0.7	55.35	62.40	7.92	10.37	7.20	28.65
0.8	56.58	60.05	7.18	11.59	13.80	**29.84**
0.9	**57.00**	55.57	5.62	4.88	17.00	28.01
1.0	52.84	42.00	6.20	**15.85**	**18.20**	27.02

Table 4: Results of the linear search for the linear weighting coefficient used in merging Llama-2-13B-based models.

α	QNLI	RTE	WNLI	Avg.
0.5	70.18	**68.23**	53.52	63.98
0.6	81.59	67.87	**57.75**	69.07
0.7	90.10	67.51	52.11	**69.91**
0.8	93.78	66.06	49.30	69.71
0.9	**94.84**	64.98	46.48	68.77
1.0	78.69	**68.23**	**57.75**	68.22

Table 5: Results of the linear search for the linear weighting coefficient used in merging LoRA adapters.

k	QNLI	RTE	WNLI	Avg.
0.5	92.00	65.34	50.70	69.35
0.6	92.84	**66.06**	49.30	69.40
0.7	93.54	64.98	52.11	70.21
0.8	94.07	64.62	**53.52**	70.74
0.9	94.51	64.26	**53.52**	**70.76**
1.0	**94.82**	63.18	52.11	70.04

Table 6: Results of selecting different values of k in LoRA merging for our FroM method.

Merging Methods	IFEval	GSM8K	HumanEval	DirectHarm4	Avg.
Baseline					
/	21.34	38.36	35.37	57.75	38.21
Fine-tuning					
/	41.13	42.76	45.73	93.00	55.66
3-Model Merging					
Task Arithmetic	**44.48**	48.29	44.51	/	45.76
TIES-Merging	**44.48**	39.42	43.90	/	42.60
DARE	26.38	0.00	0.00	/	8.79
DARE+TIES-Merging	26.38	0.00	0.00	/	8.79
RegMean	43.65	**52.08**	45.12	/	**46.95**
FroM	**44.48**	49.89	**46.34**	/	46.90
DARE+FroM	37.05	33.74	2.44	/	24.41
4-Model Merging					
Task Arithmetic	43.17	51.55	36.59	71.25	50.64
TIES-Merging	43.65	44.73	**45.73**	64.50	49.65
DARE	26.38	0.00	0.00	70.50	24.22
DARE+TIES-Merging	26.38	0.00	0.00	67.25	23.41
RegMean	43.17	**54.97**	42.07	**97.75**	**59.49**
FroM	**45.20**	52.01	45.12	65.25	51.90
DARE+FroM	21.94	32.68	0.61	72.50	31.93

Table 7: Merging results of three and four models based on Meta-LLaMA-3-8B, in comparison with RegMean.

α	IFEval	GSM8K	HumanEval	DirectHarm4	Avg.
3-Model Merging					
0.5	43.88	**49.96**	43.90	/	**45.91**
0.6	**44.48**	48.29	**44.51**	/	45.76
0.7	41.25	47.38	32.93	/	40.52
0.8	41.61	41.77	20.73	/	34.70
0.9	40.41	37.83	9.15	/	29.13
1.0	37.17	33.36	2.44	/	24.32
4-Model Merging					
0.5	**43.65**	**52.08**	**42.07**	64.25	50.51
0.6	43.17	51.55	36.59	71.25	**50.64**
0.7	39.21	49.51	23.17	**74.50**	46.60
0.8	39.57	46.47	7.93	73.50	41.87
0.9	29.14	43.06	0.61	64.75	34.39
1.0	21.94	32.68	0.61	72.75	32.00

Table 8: Results of the linear search for the weighting coefficient used for additional comparison with RegMean.

k	IFEval	GSM8K	HumanEval	DirectHarm4	Avg.
3-Model Merging					
0.5	**44.48**	**49.89**	46.34	/	**46.90**
0.6	**44.48**	48.07	45.73	/	46.09
0.7	43.41	47.01	**47.56**	/	45.99
0.8	43.88	48.67	45.73	/	46.09
0.9	43.17	47.76	45.73	/	45.55
1.0	43.76	46.25	43.90	/	44.64
4-Model Merging					
0.5	45.20	**52.01**	45.12	65.25	**51.90**
0.6	45.44	49.66	45.73	64.25	51.27
0.7	**46.04**	48.52	46.95	64.00	51.38
0.8	44.00	46.40	46.34	**69.50**	51.56
0.9	43.41	47.99	**48.17**	66.75	51.58
1.0	42.45	46.40	44.51	65.00	49.59

Table 9: Results of the linear search for k used for additional comparison with RegMean.