DPPA: Merging Large Language Model using Dynamic Pruning and Partition Amplification

Anonymous ACL submission

Abstract

Model merging aims to combine models with different capabilities into a single unified model, providing multiple capabilities without the necessity of retraining with the original training data. However, as distinctions between fine-tuned and base models grow, especially for large language models, current methods suffer significant performance drops, hindering true multi-domain capabilities. In this study, we propose a two-stage method, called Dynamic Pruning and Partition Amplification (DPPA), to address the challenge of merging models with significant distinctions. First, we introduce Dynamic Pruning (DP) to discover significant parameters and remove redundant ones. Subsequently, we propose Dynamic Partition Amplification (DPA) to restore the capability in the domain. Experimental results demonstrate that our approach performs outstandingly, improving model merging performance by almost 20%.

1 Introduction

Model merging, or model fusion, combines models with different capabilities into a unified model. Unlike multi-task learning, model merging requires no retraining on the original training data. On the one hand, model merging can combine models from different domains into a unified model, thereby offering multi-domain capabilities (Alonso et al., 2024). On the other hand, model merging can also fuse models trained on diverse data within the same domain, further enhancing overall domain performance (Fu et al., 2024). The challenge of model merging lies in resolving conflicts between the parameters of different models.

The significance of the parameter varies depending on the model. Minimal distinctions between fine-tuned models and their base models do not degrade the performance of merging model. The distinctions between fine-tuned models and their base models become more significant when a large amount of domain-specific data is used for tuning in fields such as mathematics (Hendrycks et al., 2021) and code (Rozière et al., 2023), or with advancements in techniques like instruction tuning (Mishra et al., 2022). These fine-tuned models achieve enhanced domain-specific performance, although increased parameter conflicts arise during model merging. However, current merging methods (Yu et al., 2023b; Yang et al., 2023a; Yadav et al., 2023b) experience significant performance drops with these fine-tuned models, rendering true multi-domain capabilities unattainable. Furthermore, because significance determination is based on the distinctions between these fine-tuned models and their base models, existing methods for measuring parameter significance (Sun et al., 2023; Frantar and Alistarh, 2023) are not effective.

In this study, we tackle the challenge of merging models with significant distinctions by introducing a two-stage method known as Dynamic Pruning and Partition Amplification (DPPA). First, we introduce Dynamic Pruning (DP) to discover significant parameters and remove redundant ones. Subsequently, we propose Dynamic Partition Amplification (DPA), to further amplify the importance of these significant parameters, thereby restoring domain capabilities. Our approach is applied to the delta parameter, which signifies the weight difference between the fine-tuned models and the base model.

Dynamic Pruning (DP) aims to discover significant parameters and remove redundant ones. A simple but effective way to measure significance is based on the magnitude of the delta parameter. Based on their significance, we adjust the pruning rate of different linear layers to retain the more crucial parameters. As illustrated in Figure 1, there are notable differences in significance between layers, and even within the same layer, different linear layers exhibit varying levels of significance. For

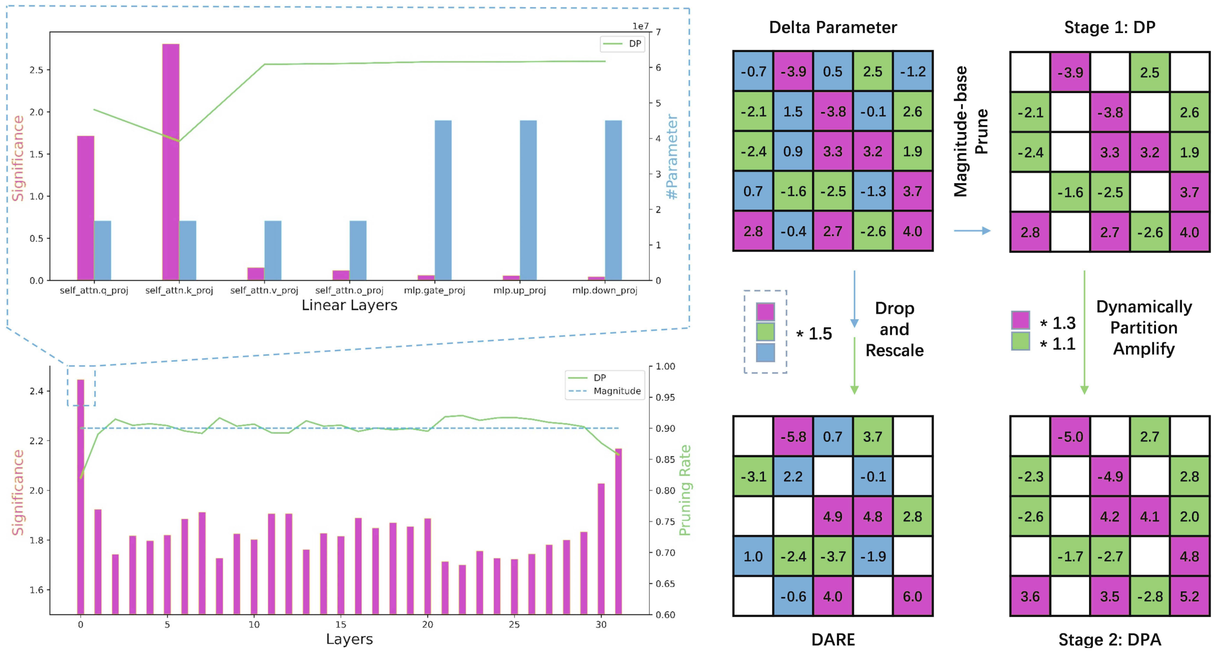

Figure 1: Within the left segment of figure, it can be found that Dynamically Pruning (DP) method modifies the pruning rate at both layer and linear layer levels, distinguishing it from magnitude pruning. On the figure's right segment, we can see the integration of DP and Dynamical Partition Amplification (DPA), paralleled with the drop and rescale operations inherent in the DARE system. This integration enhances complex model performance after the pruning process significantly.

example, the Q and K linear layers in layer *0* are more significant than the other linear layers.

Moreover, Dynamic Partition Amplification (DPA) makes these significant parameters more important to restore the capability in the domain. We discover that the necessary factor changes depending on the varying significance of the parameters. So we divide parameters based on their significance levels to get parameter partition. Each partition is then assigned various factors to enhance its domain capabilities. To evaluate the effectiveness of these factors, we use a validation dataset from the corresponding domain. As shown in Figure 1, The factors for the two partitions are *1.3* and *1.1*.

The base model adopted in this work is LLaMA-2 (Touvron et al., 2023) and Mistral (Jiang et al., 2023). We focus on two distinct domains: Mathematics and Finance. The results of the experiment show that our method only keeps 20% of parameters while yielding performance comparable to other methods that maintain up to 90% of parameters. Furthermore, our method shows outstanding performance, leading to a significant improvement of almost 20% in model merging performance. Our method even significantly outperforms others when fine-tuned models similar to the original, like Abel-mistral. In the detail analysis section, we examine the impact of ignoring parameter size and the number of parameters on performance, compare DPA with other pruning methods, and demonstrate results for different initialization methods. Through parametric analysis, we explain DPPA's effectiveness and investigate how increasing the number of domains affects model performance. We will share our code on GitHub.

2 Background

The challenge of model merging is resolving conflicts between the parameters of different models. Model merging first goes through the pruning stage, then the merging stage. For the pruning stage, the current method (Yu et al., 2023b) aims to reduce the number of conflicting parameters before parameters clash. For the merging stage, the predominant methods (Yang et al., 2023a; Yadav et al., 2023a; Jin et al., 2023) focus on resolving conflicts when parameters clash. In contrast to previous studies, our method focuses more on the pruning stage.

We review the definition of model merging and the delta parameter. It should be noted that our approach is used for the delta parameter, which represents the weight difference between the fine-tuned models and their base model.

Notation	Description
θ	a single parameter
δ, Δ	a group of parameters
S	a set of group of parameters δ
$\|\theta\|$	the absolute values of parameter θ
$\|\|\delta\|\|$	the number of parameter in group δ

Table 1: Notation system.

2.1 Model Merging Problem

Model merging combines multiple models derived from the same base model. It cannot handle the merging of multiple base models. Specifically, for models $M^1 \sim M^k$, each associated with different domains $D^1 \sim D^k$, where each domain comprises a set of tasks $D^i = \{T_1^i \sim T_n^i\}$. Here, k represents the number of domains, i represents a specific domain, and n represents the number of tasks within that domain.

By merging $M^1 \sim M^k$, we obtain the integrated model M^m, which possesses the ability to handle tasks from $D^1 \sim D^k$ simultaneously.

2.2 Delta Parameter

Analyzing the delta parameter enables a deeper understanding of the changes brought by the fine-tuning process. For each model, we find its common base model M^B and the base weight W^B. For model M^i, we have the corresponding weight W^i. We define the delta parameter as the transition of the parameter space distribution from the base model to the fine-tuned model, which is $\Delta^i = W^i - W^B$.

3 Dynamic Pruning and Partition Amplification (DPPA)

First, we introduce Dynamic Pruning (DP) to discover significant parameters and remove redundant ones. Next, we propose Dynamic Partition Amplification (DPA), which makes these significant parameters more important. Finally, we integrate the delta parameters from various fine-tuned models into the base model, resulting in a single model with multiple capabilities.

3.1 Dynamic Pruning

First, we use a single linear layer as an example to explain the overall notation system, as shown in Table 1. Next, we define the concept of parameter significance. Finally, we present the method for adjusting the pruning rate based on this significance.

For a fine-tuned model, we first get the delta parameters Δ of the model, mentioned at Sec. 2.2. We do not take into account parameters such as layer norm, focusing solely on linear layers, such as Q, K, V, O in Attention, and up/down sampling in MLP. We separate the linear layer delta parameters δ_l from Δ and denote them as S_l by

$$S_l = \{\delta_l | \delta_l \subseteq \Delta, \delta_l \text{ represents a linear layer}\}. \tag{1}$$

Parameter Significance We believe that not all parameters in the delta parameters are significant. For a group of parameters δ_l from one linear layer, the significant parameters δ_l' are N times larger than the average of absolute values of parameters δ_l by

$$\delta_l' = \{\theta' | \theta' \in \delta_l, |\theta'| > N \cdot \frac{\sum_{\theta \in \delta_l} |\theta|}{||\delta_l||}\}. \tag{2}$$

The parameter significance $sig(\cdot)$ is of the sum of the absolute values of these significant parameters to the sum of the absolute values of all parameters, as follows:

$$sig(\delta_l) = \frac{\sum_{\theta' \in \delta_l'} |\theta'|}{\sum_{\theta \in \delta_l} |\theta|}. \tag{3}$$

As demonstrated above, parameter significance primarily focuses on the values of the parameters.

Adjusting Pruning Rate Once the significance of the parameters has been determined, we can adjust the pruning rate based on the significance of various linear layers.

We translate significance to dynamic pruning rates $rat(\cdot)$ by using a modified normalization method. We consider the variations in the number of parameters among linear layers. Our goal is to ensure that the product of the adjusted pruning rates and the number of parameters in each linear layer averages out to zero, thereby maintaining the predetermined overall pruning rate. As a result, we weighted the mean significance by multiplying it with the number of parameters in each linear layer, as follows:

$$rat(\delta_l, S_l) = sig(\delta_l) - \sum_{\delta \in S_l} sig(\delta) \cdot \frac{||\delta||}{||\Delta||}. \tag{4}$$

We examine the fluctuations in adjustment rates, where excessively high adjustments have led to pruning rates exceeding 100%. We define the maximum value of pruning rate fluctuation as λ. As a

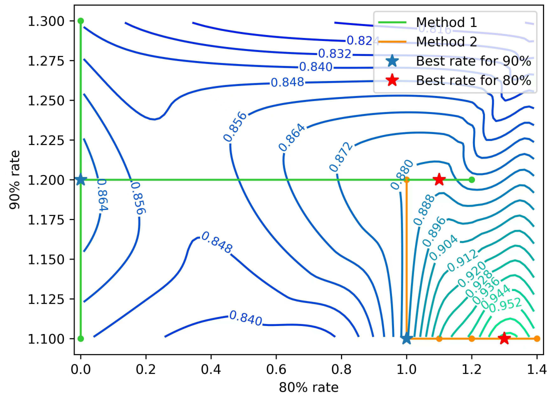

Figure 2: We use green and orange lines to show amplification rate trajectories. The blue star marks the optimal rate at 90% pruning, and the red star marks it at 80%. Contour lines illustrate performance in the mathematical domain.

result, We first find the maximum absolute value of the dynamical pruning rates, $rat(\cdot)$, across all linear layers. The scaling factor, $fac(\cdot)$, is then calculated by dividing λ by this maximum value, as illustrated below:

$$fac(S_l) = \frac{\lambda}{\arg\max_{\delta \in S_l} abs(rat(\delta, S_l))}. \quad (5)$$

Following the principle that higher parameter significance corresponds to lower pruning rates, We modify the pruning rate by applying a scaling factor, resulting in the final adjusted rate for a linear layer, α_l, as follows:

$$\alpha_l = \alpha - fac(S_l) \cdot rat(\delta_l, S_l), \quad (6)$$

where α represents predetermined overall pruning rate.

3.2 Dynamic Partition Amplification

First, we apply Dynamic Pruning at various pruning rates to partition the parameters. To restore performance, we amplify and combine these partitions. By acknowledging parameter interactions during enhancement, we propose two initialization methods and assess their effectiveness across various scenarios. Finally, we provide detailed information on the data used and the validation metrics employed during the enhancement process.

Partition of Parameters The number of retained parameters varies with different pruning rates. Compared to lower pruning rates, the higher pruning rates retained the fewer but more crucial parameters. At lower pruning rates, more parameters are retained. For example, as shown in Fig. 1, higher pruning rates retain only the purple parameters, while lower rates retain both the purple and green parameters. Therefore, the parameter partition for the lower rate includes the green parameters. We set the partition size to β, implying that when the low pruning rate is x, the high pruning rate becomes $x + \beta$.

Partition Amplification Partitions with higher pruning rates are considered more important. The importance of the partitions is ranked based on their pruning rates. After initialization, we first amplify the most important partition. By multiplying the partition parameters by a dynamic factor, an expanded partition is obtained. This dynamic factor starts at 1 and increases by a hyperparameter, denoted as γ, until optimal performance is achieved. Once the primary parameter partition factor is determined, adjust the secondary parameter partitions accordingly, and continue this process as needed.

Initialization methods There are interacted among partition parameters, and our approach only changes one partition at each stage. Thus, whether considering the impact of other partitions when amplifying partition is crucial. We propose two initialization methods: one ignoring parameter interactions and the other considering them. Use the first method if performance differences between partitions are within 5%, otherwise use the second method. **Method 1** adjust parameters within the 90% pruning rate partition, setting the remainder to zero. The resulting curve from this method is illustrated by the green line in Fig. 2. **Method 2** use the partition that matches the target pruning rate while adjusting the 90% partition. The resulting curve from this method is illustrated by the orange line in Fig. 2.

Validation Metrics For adjusted models mentioned above, we verify their capabilities using in-domain datasets. No additional training is required; we simply infer the model's performance on the validation dataset.

To normalize performance differences across tasks, we introduced the Task-Ratio metric. For a task T_j, the Task-Ratio is the performance ratio of the adjusted model M_{adj} to the dense model

M_{den}, defined as:

$$\text{Task-Ratio}_j = \frac{Per(M_{adj}, T_j)}{Per(M_{den}, T_j)}, \quad (7)$$

where $Per(M, T)$ represents the performance of model M on task T. According to the formula, the Task-Ratio of the dense model is 100%.

We propose Domain-Ratio metrics to evaluate performance across abundant datasets in a domain. We use a multiplicative approach to account for all tasks and avoid obscuring low-performance ones. To make performance independent of task number n, we square the product. The formula for Domain-Accuracy is as follows:

$$\text{Domain-Ratio} = \sqrt[n]{\Pi_{j=1}^{n} \text{Task-Ratio}_j}. \quad (8)$$

3.3 Model Merging

After applying Dynamic Pruning and Partition Amplification, We obtained the pruned delta parameters of different models. In Section 5.3, we refer to multiple existing methodologies for merging stage. We employ Ties-Merging (Yadav et al., 2023b), to resolve parameter conflicts during the merging stage after the pruning stage. Thus, we get the final merging model:

$$W^m = W^B + \text{Ties}(\Sigma_{i=1}^{k} \text{DPPA}(\Delta^i)) \quad (9)$$

4 Experiments

4.1 Experimental Setup

Pre-Trained Backbone and Fine-tune Models Considering the need to fine-tune the base model for different domains and its performance impact, we chose LLaMa 2 (Touvron et al., 2023) as the base model over other pre-trained models. For the mathematics and finance domains, we selected two high-performing models: Abel (Chern et al., 2023) and Finance-chat (Cheng et al., 2023). We chose Mistral despite its few fine-tuning models to test our method on different base models and minimal variations from the original. Abel-Mistral represents such small differences.

Datasets and Metric For each domain, we selected two datasets. In mathematics, we chose GSM8k (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021), evaluating models using zero-shot accuracy with Abel's testing script (Chern et al., 2023). In finance, we chose FiQA_SA (Maia et al., 2018) and FPB (Malo et al., 2014), also using zero-shot accuracy. For AdaptLLM (Cheng et al., 2023), without a testing script, we deemed a multiple-choice question correct if the predicted sentence included the correct choice. The evaluation metric is detailed in Sec. 3.2.

Implementation Details In our study using the vLLM framework, we set a batch size of 32 for GSM8k and MATH, and a batch size of 1 for FiQA_SA and FPB. We used greedy decoding with a temperature of 0 and a maximum generation length of 2048, conducted on an NVIDIA Tesla A100 GPU. We set N to 5, λ to 0.08, and both β and γ to 0.1.

4.2 Baseline Method

We establish two sample weight averaging methods, one merging-based, and five pruning-based methods as baselines. they are described below:

- **Model Soups** (Wortsman et al., 2022) averages all model parameters.
- **LM-Cocktail** (Xiao et al., 2023) weights models from different domains to select the optimal result.
- **Ties-Merging** (Yadav et al., 2023b) resolves parameter conflicts during merging stage.
- **Wanda** (Sun et al., 2023) trims parameters that minimally impact inference.
- **SparseGPT** (Frantar and Alistarh, 2023) adjusts pruned parameters for better performance.
- **Magnitude** (Han et al., 2015b) keeps weights with larger absolute values, removing smaller ones.
- **OWL** (Yin et al., 2023) recognizes parameter significance varies across model layers.
- **DARE** (Yu et al., 2023b) starts with random pruning, then expands remaining parameters based on pruning rate.

4.3 Main Result of DPPA

We present the Domain-Ratio and Task-Ratio results for all datasets. Table 2 displays results for three models with varying pruning rates. Our method performs optimally at high pruning rates on both Llama2 and Mistral, regardless of Domain-Ratio or Task-Ratio. The experimental results show our approach retains only 20% of parameters yet performs comparably to methods retaining 90%,

Sparse Ratio	Domain-Ratio				Task	Task-Ratio			
	Magnitude	OWL	DARE	DPPA		Magnitude	OWL	DARE	DPPA
Abel-Llama									
10%	96.46	96.69	96.64	**98.86**	GSM8k	**100.14**	99.63	98.23	98.49
					Math	92.92	93.84	95.07	**99.23**
80%	80.12	77.11	87.41	**97.08**	GSM8k	83.78	82.77	89.49	**95.56**
					Math	76.61	71.84	85.38	**98.61**
90%	53.41	54.09	73.44	**86.85**	GSM8k	57.42	57.29	83.28	**87.71**
					Math	49.69	51.07	64.76	**86.00**
Finance-Llama									
10%	90.81	89.12	91.04	**97.05**	FiQA_SA	88.81	86.95	91.92	**95.14**
					FPB	92.84	91.35	90.16	**99.01**
80%	71.04	74.92	84.01	**96.65**	FiQA_SA	75.77	81.36	83.22	**94.41**
					FPB	66.61	69.00	84.79	**98.95**
90%	54.71	56.74	82.90	**92.11**	FiQA_SA	53.41	57.76	83.85	**88.82**
					FPB	56.03	55.73	81.96	**95.52**
Abel-Mistral									
10%	99.63	99.67	**99.75**	99.70	GSM8k	99.82	99.82	**99.85**	99.82
					Math	99.45	99.52	**99.66**	99.59
80%	93.46	92.52	95.32	**99.98**	GSM8k	92.50	92.31	94.72	**97.38**
					Math	94.43	92.73	95.92	**102.64**
90%	81.24	79.92	86.88	**94.99**	GSM8k	84.90	83.49	88.66	**93.15**
					Math	77.73	76.51	85.13	**96.87**

Table 2: Domain-Ratio and Task-Ratio of different methods at various pruning rates. Additional results under remainder pruning rates and the specific performance values for different tasks are presented in Appendix A.

Methods	Math	Fin	Average
Model Soups	15.99	79.46	47.73
LM-Cocktail	76.96	78.80	77.88
Ties-Merging	**96.23**	22.12	59.18
w/ Wanda	8.30	20.65	14.48
w/ SparseGPT	21.74	18.60	20.17
w/ DARE 90%	21.10	64.88	42.99
w/ DPPA 90%	89.25	79.40	84.33
w/ DARE 80%	58.43	77.16	67.79
w/ DPPA 80%	92.75	**95.45**	**94.10**

Table 3: Domain-Ratio of the merged Llama model that combines domains mathematics and finance. The specific performance values are presented in Appendix A.

Domains	Magnitude	OWL	DP
Math	53.41	54.09	**54.97**
Fin	54.71	56.74	**62.06**

Table 4: Domain-Ratio of DP at a pruning rate of 90%.

guaranteeing over 96% of the domain's performance.

Due to space constraints, detailed values, remainder pruning rates, and DPA parameter partition factors are included in Appendix A.

4.4 Main Result of Merge Methods

We validate our pruning method for model merging by integrating models. Table 3 displays results of two domains at 80% and 90% pruning rates and other baselines. Sample weight averaging methods like Model Soups and LM-Cocktail suffer performance degradation due to unresolved parameter conflicts. Traditional pruning methods like Wanda and SparseGPT measures the importance of full parameter, unlike the delta parameter, impacting the model after merging. Our method improves performance by over 20% compared to DARE at the same pruning rate, demonstrating its efficacy in model merging.

Model	Min	10%	90%	Max
Abel-Llama	-0.01733	-0.00114	0.00114	0.02014
Fin-Llama	-0.02612	-0.00160	0.00160	0.02011
Abel-Mistral	-0.00127	-0.00010	0.00010	0.00139

Table 5: The offset of different models from the base model at different position proportions.

4.5 Detail Analysis

We present the performance of DP in Table 4 and discuss cases where DP can replace DARE. Table 6 examines the results of disregarding the parameter magnitude considering only the number of parameters as the definition of parameter significance and the effects of rounding off $fac(\cdot)$. We compare performance of DPA using other pruning methods in Table 7 and demonstrate the performance of two

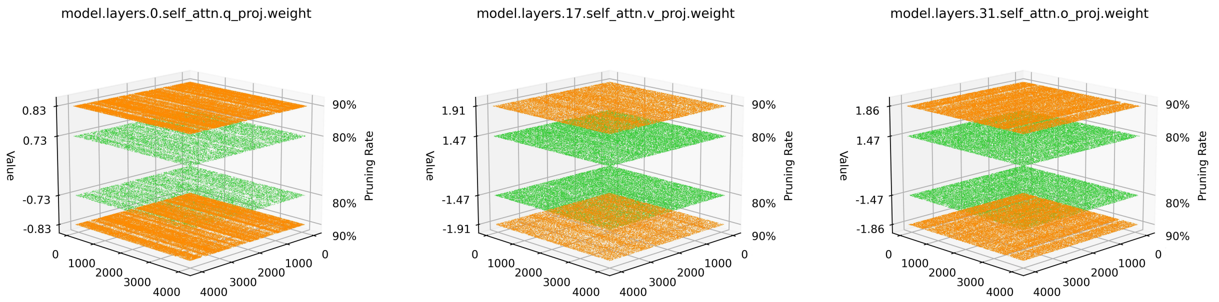

Figure 3: After analyzing the pruned parameters of the financial model, it is evident that there is a higher parameter count in the initial and final 0, 31 layers, while the middle 17 layers have fewer parameters. Additionally, in the Q, K, V components, it is observed that 90% of the parameters are concentrated in certain dimensions. To facilitate observation, we have amplified the value by a factor of 1000.

Methods	Math	Fin
DP	**54.97**	**62.06**
change_sig	53.13	60.57
w/o fac	52.69	61.84

Table 6: Domain-Ratio of the variants of DP at a pruning rate of 90%.

Methods	Math	Fin
DPPA	**86.85**	**92.11**
DARE	73.44	82.90
w/DPA	83.63	85.08
OWL	54.09	56.74
w/DPA	84.24	87.56

Table 7: Domain-Ratio of DARE and OWL using DPA at a pruning rate of 90%.

Domains	Method 1	Method 2
Math	88.45	**97.08**
Fin	**96.65**	94.89

Table 8: Domain-Ratio of two method in DPA at a pruning rate of 80%.

different initializations in Table 8. We analyzed why DPPA is effective, as shown in the Fig. 3. Finally, we explore the performance impact of adding a domain in Table 9.

The Effectiveness of DP As seen in Table 4, DP outperforms at high pruning rates by adjusting the significance of parameters within each layer, retaining crucial ones. The DARE method struggles when parameter deviations exceed 0.03, with performance worsening as offsets increase (see Table 5). More detailed results are in Appendix B. When DARE's performance drops below 90% at a 90% pruning rate, our method offers a viable alternative.

The Variants of DP As shown in Table 6, change_sig disregards parameter magnitude, considering only the number of parameters for significance, while w/o fac ignores effects of $fac(\cdot)$. Removing the parameter importance causes a significant performance drop, while the tuning factor has a minor effect.

The Generality of DPA Our experimental results are in Table 7. We tested the DPA method on DARE and OWL. Since DARE already amplifies parameters significantly at high pruning rates (5x for 80% and 10x for 90%), we switched to dynamic reduction. Since Owl is similar to the DP method, its performance with DPA surpasses DARE's.

Initialization methods We show a performance comparison of the two initialization methods at 80% pruning rate in Table 8. For models with small performance differences, use method 1; for large differences, use method 2, which offers more significant improvement.

why DPPA is effective? To investigate, we analyzed the Delta parameters (see Fig 3), exploring the relationship between remaining parameters after DP at different pruning rates and linear layers. The graph shows that, despite being an unstructured pruning method, DP exhibits aspects of structured pruning at high pruning rates. This dimension partitioning aids in interpreting parameter space distribution within specific domains. Using DPA, we amplify parameters, strengthen domain-specific weights in these dimensions, and restore certain capabilities.

Method & Pruning Rate	Math	Fin	Law
DARE 90%	7.89	51.48	53.86
DPPA 90%	89.95	85.24	122.08
DARE 80%	32.61	74.49	78.11
DPPA 80%	**91.28**	**95.20**	**146.23**

Table 9: Domain-Ratio of the model that combines domains mathematics, finance and law.

Mergeing more Domain In Table 9, we present the merging results for adding law domains. Comparing this with Table 3, it is evident that integrating a fine-tuned model from an additional domain greatly degrades DARE's performance. Conversely, our method maintains comparable performance despite the extra domain, though performance decreases at varying pruning rates. This result is expected, as parameter conflicts during model merging typically cause performance degradation. Relevant information about the added law domain is placed in Appendix C.

5 Related Work

5.1 Pruning Techniques

Traditional pruning techniques aim to reduce model parameters (Zhu et al., 2023). Although extensively studied (Hubara et al., 2021; Mozer and Smolensky, 1988; Han et al., 2015a; Lin et al., 2019), progress has been slow with large language models due to the significant fine-tuning data required. LORA fine-tuning (Ma et al., 2023) was proposed to restore performance. Newer methods avoid fine-tuning: SparseGPT (Frantar and Alistarh, 2023) uses the Hessian matrix for pruning and weight updates to reduce reconstruction error, Wanda (Sun et al., 2023) combines weight magnitudes and input activations, DSOT (Zhang et al., 2023c) adjusts parameters to minimize discrepancies, and OWL (Yin et al., 2023) introduces non-uniform layered sparsity for higher pruning rates.

5.2 Special Domain Fine-Tuning

This trend continues with large language models, leading to domain-specific models in fields like coding (Rozière et al., 2023; Yu et al., 2023c; Luo et al., 2023b), mathematics (Luo et al., 2023a; Yue et al., 2023; Yu et al., 2023a; Gou et al., 2023; Yuan et al., 2023), medicine (Kweon et al., 2023; Chen et al., 2023; Toma et al., 2023), and finance (Zhang et al., 2023a; Yang et al., 2023b; Xie et al., 2023). However, fine-tuning across multiple domains demands significant computational resources, prompting interest in model merging methods.

5.3 Model Merge

Model merging methods include alignment (Li et al., 2016), model ensemble (Pathak et al., 2010), module connection (Freeman and Bruna, 2017), and weight averaging (Wang et al., 2020). Of these, only weight averaging reduces model parameters. Approaches within weight averaging include subspace weight averaging (Li et al., 2023), SWA (Izmailov et al., 2018), and task arithmetic (Ilharco et al., 2023). Task arithmetic is notable as it involves domain-specific offsets added or subtracted from base model weights. Further developments in task arithmetic focus on LORA (Zhang et al., 2023b; Chitale et al., 2023; Chronopoulou et al., 2023) and minimizing parameter conflicts via scaling coefficients (Ortiz-Jiménez et al., 2023; Yang et al., 2023a; Yadav et al., 2023b; Stoica et al., 2023), selective weight retention (Yadav et al., 2023a), and vector space adjustments (Jin et al., 2023).

5.4 Federated Learning

Federated learning allows multiple clients to collaboratively train models under a central aggregator, preserving data privacy (Zhang et al., 2021). This setup aligns well with model merging, as it combines locally trained models without risking data leakage.

6 Conclusions

In this study, we introduce a pruning method called DP, which is an improved approach based on magnitude pruning to enhance performance at higher pruning rates. Subsequently, we propose DPA, which focuses on dynamically amplifying partitions of parameters based on their varying levels of significance. Using DPPA, we address the challenge of model merging in complex fine-tuned models. The experimental results show that our approach only keep 20% of the specific domain parameters, while achieves comparable performance to other methods that retain 90% of the specific domain parameters. Furthermore, our method also achieves a significant improvement of nearly 20% in model merging. Through parametric analysis, we explain DPPA's effectiveness and investigate how increasing the number of domains affects model performance.

Limitations

Our method performs less effectively than DARE on fine-tuned models with minimal differences compared to the original model.

DAP requires a longer time to find the optimal ratio.

While it mitigates parameter conflicts in model merging, there remains the issue of performance degradation.

References

Iñigo Alonso, Maite Oronoz, and Rodrigo Agerri. 2024. Medexpqa: Multilingual benchmarking of large language models for medical question answering. *CoRR*, abs/2404.05590.

Zeming Chen, Alejandro Hernández-Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, Alexandre Sallinen, Alireza Sakhaeirad, Vinitra Swamy, Igor Krawczuk, Deniz Bayazit, Axel Marmet, Syrielle Montariol, Mary-Anne Hartley, Martin Jaggi, and Antoine Bosselut. 2023. MEDITRON-70B: scaling medical pretraining for large language models. *CoRR*, abs/2311.16079.

Daixuan Cheng, Shaohan Huang, and Furu Wei. 2023. Adapting large language models via reading comprehension. *CoRR*, abs/2309.09530.

Ethan Chern, Haoyang Zou, Xuefeng Li, Jiewen Hu, Kehua Feng, Junlong Li, and Pengfei Liu. 2023. Generative ai for math: Abel. `https://github.com/GAIR-NLP/abel`.

Rajas Chitale, Ankit Vaidya, Aditya Kane, and Archana Ghotkar. 2023. Task arithmetic with lora for continual learning. *CoRR*, abs/2311.02428.

Alexandra Chronopoulou, Jonas Pfeiffer, Joshua Maynez, Xinyi Wang, Sebastian Ruder, and Priyanka Agrawal. 2023. Language and task arithmetic with parameter-efficient layers for zero-shot summarization. *CoRR*, abs/2311.09344.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 10323–10337. PMLR.

C. Daniel Freeman and Joan Bruna. 2017. Topology and geometry of half-rectified network optimization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tingchen Fu, Deng Cai, Lemao Liu, Shuming Shi, and Rui Yan. 2024. Disperse-then-merge: Pushing the limits of instruction tuning via alignment tax reduction. *arXiv preprint arXiv:2405.13432*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *CoRR*, abs/2309.17452.

Song Han, Jeff Pool, John Tran, and William J. Dally. 2015a. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1135–1143.

Song Han, Jeff Pool, John Tran, and William J. Dally. 2015b. Learning both weights and connections for efficient neural networks. *CoRR*, abs/1506.02626.

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. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Itay Hubara, Brian Chmiel, Moshe Island, Ron Banner, Joseph Naor, and Daniel Soudry. 2021. Accelerated sparse neural training: A provable and efficient method to find N: M transposable masks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 21099–21111.

Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pages 876–885. AUAI Press.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Sunjun Kweon, Junu Kim, Jiyoun Kim, Sujeong Im, Eunbyeol Cho, Seongsu Bae, Jungwoo Oh, Gyubok Lee, Jong Hak Moon, Seng Chan You, Seungjin Baek, Chang Hoon Han, Yoon Bin Jung, Yohan Jo, and Edward Choi. 2023. Publicly shareable clinical large language model built on synthetic clinical notes. *CoRR*, abs/2309.00237.

Tao Li, Lei Tan, Zhehao Huang, Qinghua Tao, Yipeng Liu, and Xiaolin Huang. 2023. Low dimensional trajectory hypothesis is true: Dnns can be trained in tiny subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(3):3411–3420.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John E. Hopcroft. 2016. Convergent learning: Do different neural networks learn the same representations? In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David S. Doermann. 2019. Towards optimal structured CNN pruning via generative adversarial learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2790–2799. Computer Vision Foundation / IEEE.

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

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023b. Wizardcoder: Empowering code large language models with evol-instruct. *CoRR*, abs/2306.08568.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *CoRR*, abs/2305.11627.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: Financial opinion mining and question answering. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1941–1942. ACM.

Pekka Malo, Ankur Sinha, Pekka J. Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *J. Assoc. Inf. Sci. Technol.*, 65(4):782–796.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3470–3487. Association for Computational Linguistics.

Michael Mozer and Paul Smolensky. 1988. Skeletonization: A technique for trimming the fat from a network via relevance assessment. In *Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]*, pages 107–115. Morgan Kaufmann.

Guillermo Ortiz-Jiménez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. *CoRR*, abs/2305.12827.

Manas A. Pathak, Shantanu Rane, and Bhiksha Raj. 2010. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 1876–1884. Curran Associates, Inc.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950.

George Stoica, Daniel Bolya, Jakob Bjorner, Taylor Hearn, and Judy Hoffman. 2023. Zipit! merging models from different tasks without training. *CoRR*, abs/2305.03053.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2023. A simple and effective pruning approach for large language models. *CoRR*, abs/2306.11695.

Augustin Toma, Patrick R. Lawler, Jimmy Ba, Rahul G. Krishnan, Barry B. Rubin, and Bo Wang. 2023. Clinical camel: An open-source expert-level medical language model with dialogue-based knowledge encoding. *CoRR*, abs/2305.12031.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris S. Papailiopoulos, and Yasaman Khazaeni. 2020. Federated learning with matched averaging. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Xingrun Xing. 2023. Lm-cocktail: Resilient tuning of language models via model merging. *CoRR*, abs/2311.13534.

Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. 2023. PIXIU: A large language model, instruction data and evaluation benchmark for finance. *CoRR*, abs/2306.05443.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023a. Resolving interference when merging models. *CoRR*, abs/2306.01708.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. 2023b. Ties-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2023a. Adamerging: Adaptive model merging for multi-task learning. *CoRR*, abs/2310.02575.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. 2023b. Fingpt: Open-source financial large language models. *CoRR*, abs/2306.06031.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Mykola Pechenizkiy, Yi Liang, Zhangyang Wang, and Shiwei Liu. 2023. Outlier weighed layerwise sparsity (OWL): A missing secret sauce for pruning llms to high sparsity. *CoRR*, abs/2310.05175.

Fei Yu, Anningzhe Gao, and Benyou Wang. 2023a. Outcome-supervised verifiers for planning in mathematical reasoning. *CoRR*, abs/2311.09724.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023b. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *CoRR*, abs/2311.03099.

Zhaojian Yu, Xin Zhang, Ning Shang, Yangyu Huang, Can Xu, Yishujie Zhao, Wenxiang Hu, and Qiufeng Yin. 2023c. Wavecoder: Widespread and versatile enhanced instruction tuning with refined data generation. *CoRR*, abs/2312.14187.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *CoRR*, abs/2308.01825.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *CoRR*, abs/2309.05653.

Boyu Zhang, Hongyang Yang, and Xiao-Yang Liu. 2023a. Instruct-fingpt: Financial sentiment analysis by instruction tuning of general-purpose large language models. *CoRR*, abs/2306.12659.

Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. 2021. A survey on federated learning. *Knowl. Based Syst.*, 216:106775.

Jinghan Zhang, Shiqi Chen, Junteng Liu, and Junxian He. 2023b. Composing parameter-efficient modules with arithmetic operations. *CoRR*, abs/2306.14870.

Yuxin Zhang, Lirui Zhao, Mingbao Lin, Yunyun Sun, Yiwu Yao, Xingjia Han, Jared Tanner, Shiwei Liu, and Rongrong Ji. 2023c. Dynamic sparse no training: Training-free fine-tuning for sparse llms. *CoRR*, abs/2310.08915.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2023. A survey on model compression for large language models. *CoRR*, abs/2308.07633.

A Main Result of Various Pruning Methods on Specific Tasks

We presented all pruning results of Llama-based model in Table 13 and Mistral-based model in Table 11. The table displays the performance of two

Model	Min	10%	20%	30%	40%	50%	60%	70%	80%	90%	Max
Abel-Llama	-0.0173	-0.0011	-0.0007	-0.0004	-0.0002	1.1e-08	0.0002	0.0004	0.0007	0.0011	0.0201
Finance-Llama	-0.0261	-0.0016	-0.0010	-0.0006	-0.0003	0.0	0.0003	0.0006	0.0010	0.0016	0.0201
Abel-Mistral	-0.0012	-0.0001	-7.1e-05	-4.4e-05	-2.1e-05	-5.8e-10	2.1e-05	4.4e-05	7.1e-05	0.0001	0.0013

Table 10: The offset of different models from the base model at different position proportions.

Sparse ratio	Magnitude	OWL	DP	DARE
gsm8k				
0.1	0.806671721	0.806671721	0.804397271	0.806887854
0.2	0.806671721	0.805155421	0.803639121	0.805155421
0.3	0.805155421	0.808188021	0.808188021	0.806671721
0.4	0.806671721	0.807429871	0.808188021	0.803639121
0.5	0.794541319	0.80288097	0.79681577	0.805913571
0.6	0.785443518	0.782410917	0.784685368	0.809704321
0.7	0.761182714	0.762699014	0.760424564	0.780136467
0.8	0.747536012	0.746019712	0.746777862	0.765432321
0.9	0.686125853	0.674753601	0.683093252	0.716461463
MATH				
0.1	0.2930	0.2932	0.2930	0.2936
0.2	0.2916	0.2916	0.2910	0.2924
0.3	0.2938	0.2936	0.2926	0.2944
0.4	0.2982	0.2964	0.2968	0.2932
0.5	0.2948	0.2954	0.2946	0.2966
0.6	0.2900	0.2950	0.2934	0.2958
0.7	0.2866	0.2876	0.2902	0.2914
0.8	0.2782	0.2732	0.2746	0.2826
0.9	0.2290	0.2254	0.2250	0.2508

Table 11: All pruning result for Abel-Mistral model in math domain.

llama2-based models in their respective domains, including DP performance and DPA search results in various domains.

We show the factor of DPA and the corresponding results on each dataset. For Abel-Llama, the amplification factor is 1.3 for 80% and 1.1 for 90% of the partitions; for gsm8k is 0.5716, for Math is 0.1282. For Finance-Llama, the factor is 1.0 for 80% and 1.1 for 90% of the partitions; for FiQA_SA is 0.646808511, for FPB is 0.684536082. For Abel-Mistral, the factor is 1.0 for 80% and 1.7 for 90% of the partitions; for gsm8k is 0.7870, for Math is 0.3024.

And, we show the numerical results after the Merging of each method as shown in the Table 12.

B The Offset of Models

We presented ten different percentage values in Table 10.

C Law

Our method achieves performance close to the dense model but may fall short for tasks requiring superior performance. Interestingly, in the law domain, pruned models significantly outperformed the dense model, achieving 120-140% of its performance at pruning rates of 10-90%. We attribute this to the low performance of the law domain fine-tune model and the potential enhancement from offsetting a local minimum through pruning.

Methods	GSM8k	MATH	FiQA_SA	FPB
Model Soups	0.121304018	0.0164	0.544680851	0.549484536
LM-Cocktail	0.473843821	0.0972	0.527659574	0.557731959
Ties-Merging	0.576952236	0.1248	0.208510638	0.111340206
w/ Wanda	0.039423805	0.0136	0.132471678	0.169123487
w/ SparseGPT	0.062816479	0.0528	0.12158879	0.134876196
w/ DARE 90%	0.154662623	0.0224	0.455319149	0.43814433
w/ DPPA 90%	0.557998484	0.111	0.591489362	0.505154639
w/ DARE 80%	0.392721759	0.0676	0.523404255	0.539175258
w/ DPPA 80%	0.577710387	0.1158	0.663829787	0.650515464

Table 12: The specific performance values of the merged Llama model that combines domains mathematics and finance.

Sparse ratio	Magnitude	OWL	DP	DARE
gsm8k				
0.1	0.59893859	0.595905989	0.589082638	0.587566338
0.2	0.593631539	0.592873389	0.59893859	0.585291888
0.3	0.590598939	0.589082638	0.594389689	0.586808188
0.4	0.578468537	0.579984837	0.588324488	0.567096285
0.5	0.584533738	0.589840788	0.587566338	0.563305534
0.6	0.578468537	0.574677786	0.570128886	0.557240334
0.7	0.546626232	0.542835481	0.545109932	0.558756634
0.8	0.501137225	0.495072024	0.489006823	0.53525398
0.9	0.343442002	0.342683851	0.351781653	0.498104625
MATH				
0.1	0.1208	0.122	0.129	0.1236
0.2	0.1218	0.1212	0.1232	0.1298
0.3	0.125	0.1232	0.1238	0.1274
0.4	0.1262	0.1258	0.1276	0.1264
0.5	0.122	0.125	0.1248	0.1216
0.6	0.1254	0.124	0.1194	0.1184
0.7	0.1176	0.1148	0.1142	0.1134
0.8	0.0996	0.0934	0.095	0.111
0.9	0.0646	0.0664	0.0668	0.0842
FiQA_SA				
0.1	0.608510638	0.595744681	0.635744681	0.629787234
0.2	0.612765957	0.642553191	0.629787234	0.621276596
0.3	0.629787234	0.646808511	0.621276596	0.634042553
0.4	0.629787234	0.621276596	0.629787234	0.625531915
0.5	0.582978723	0.561702128	0.34893617	0.561702128
0.6	0.595744681	0.540425532	0.54893617	0.685106383
0.7	0.540425532	0.510638298	0.195744681	0.587234043
0.8	0.519148936	0.557446809	0.493617021	0.570212766
0.9	0.365957447	0.395744681	0.438297872	0.574468085
FPB				
0.1	0.642268041	0.631958763	0.62556701	0.62371134
0.2	0.620618557	0.616494845	0.611340206	0.634020619
0.3	0.597938144	0.608247423	0.628865979	0.627835052
0.4	0.610309278	0.609278351	0.601030928	0.644329897
0.5	0.590721649	0.57628866	0.605154639	0.611340206
0.6	0.597938144	0.579381443	0.579381443	0.615463918
0.7	0.534020619	0.550515464	0.537113402	0.607216495
0.8	0.460824742	0.477319588	0.471134021	0.586597938
0.9	0.387628866	0.38556701	0.416494845	0.567010309

Table 13: All pruning result for Llama-based model in two domain.