DPPA: Pruning Method for Large Language Model to Model Merging

Anonymous ACL submission

Abstract

Model merging is the process of combining models from various domains into a single model with multi-domain capabilities, and the challenge is to resolve parameter conflicts. To reduce the possibility of parameter conflicts, the pruning method is used to remove parameters from a model. The recent method utilizes a domain-independent pruning technique which is based on the assumption that there is little variation between different model parameters. We found that because domain-independent methods remove some domain-specific parameters, they are ineffective when there are significant distinctions in model parameters. In this paper, we address the challenge of merging models with significant distinctions by proposing a two-stage method called DPPA. First, we introduce Dynamically Pruning (DP) to discover domain-specific significant parameters and remove redundant ones. Subsequently, to enhance the capability in the domain, we propose Dynamical Partition Amplification (DPA), which amplifies significant parameters during the merging process. The results of the experiments demonstrate that our approach performs outstandingly, improving model merging performance by almost 20%. We will share our code on GitHub.

1 Introduction

Model merging, referred to as model fusion, is a method that merges models from diverse domains into a single model with multi-domain capabilities. The challenge in this task is how to resolve parameter conflicts. On one hand, the predominant methods (Yang et al., 2023a; Yadav et al., 2023; Jin et al., 2023) focus on dealing with conflicting parameters in the merging stage. On the other hand, to reduce the possibility of parameter conflicts, the pruning method is used to remove parameters from a model.

The recent method (Yu et al., 2023b) utilizes a domain-independent pruning technique which is based on the assumption that there is little variation between different model parameters. Exceptional results have been achieved in situations with little model differences. With the development of training techniques and data, the difference between state-of-the-art models and base models in various domains is becoming increasingly significant. However, utilizing existing methods to merge complex models causes significant performance degradation. We found that because domain-independent methods remove some domain-specific parameters, they are ineffective when there are significant distinctions in model parameters.

In this paper, we address the challenge of merging models with significant distinctions by proposing a two-stage method called DPPA. First, we introduce Dynamically Pruning (DP) to discover domain-specific significant parameters and remove redundant ones. Subsequently, to enhance the capability in the domain, we propose Dynamical Partition Amplification (DPA), which amplifies significant parameters during the merging process. It is noted that our approach is used for the delta parameter difference between the fine-tuned model and the base model.

Dynamically Pruning (DP) is employed to adjust the pruning rate based on the significance of different linear layers. A simple and effective way to measure significance is based on the magnitude of the parameter. OWL (Yin et al., 2023) observes that the significance of parameters varies across different layers. We believe in scenarios at high pruning rates, it is important to enhance the refinement of the parameter's significance and modify the pruning rate at the linear layers level. For example, As illustrated in Figure 1, it is apparent that the Q and K linear layers in layer 0 hold more significant values when compared to other linear layers. Our approach considers the linear layer (such as Q, K,

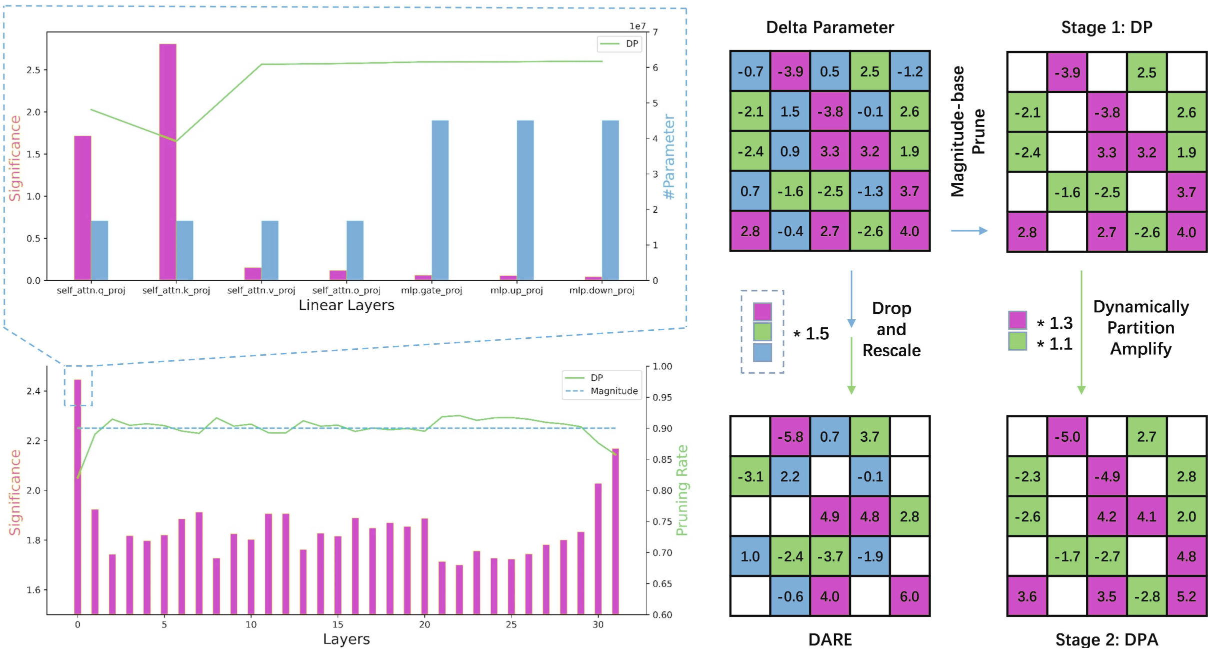

Figure 1: Within the figure's left segment, it is visible that Dynamically Pruning (DP) method modifies the pruning rate at both layer and linear layer levels, distinguishing it from magnitude pruning. On the figure's right segment, we can see the integration of DP and Dynamical Partition Amplification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

V, O in Attention and up/down sampling in MLP) as the minimum unit for adjusting pruning rates and modifies these rates based on the significance of the parameters.

Moreover, Dynamical Partition Amplification (DPA) is a rescaling method that dynamically amplifies partitions of parameters based on the varying significance of the parameters. It is built upon the pruning approach. Firstly, we partition parameters according to different degrees of significance. Secondly, considering the interactive influence between parameters, we employ two methods of initialization. Lastly, we prioritize amplifying parameters of high significance in the order of their significance. We adopt the initialization method with superior performance as our final result.

The base model we employ in our paper is LLaMA 2 (Touvron et al., 2023b). We focus on three distinct domains: Mathematics, Finance, and Law. The results of the experiment show that our method only keeps 20% of domain-specific parameters while yielding performance comparable to other methods that maintain up to 90% of parameters. This demonstrates that our method removes redundancy and maintains domain-specific parameters effectively. Furthermore, our method displays outstanding performance, leading to a significant improvement of nearly 20% in model merging performance. We conduct experiments in the scenarios of both three-domain and two-domain merging, and the results show that the impact of the extra domain on our approach is essentially insignificant. We further substantiate the viability of DPA on other pruning methods. Although it doesn't yield a level of performance equal to DPPA, it moderately enhances performance.

2 Related Work

2.1 Pruning Technique

Traditional pruning techniques aim to reduce the number of parameters in a model (Zhu et al., 2023). There have been several studies conducted on this topic, both in the era of pre-trained language models and before (Hubara et al., 2021; Mozer and Smolensky, 1988; Han et al., 2015a; Lin et al., 2019). However, progress in these studies has been relatively slow in the era of large language models, as pruning requires a substantial amount of data for fine-tuning, which is not feasible for such models. To tackle this issue, LORA fine-tuning was proposed by Ma et al. (2023) to restore the original performance. Recently, some studies have shifted their focus to pruning methods that do not necessitate fine-tuning. For instance, SparseGPT (Frantar and Alistarh, 2023) utilizes the Hessian matrix for pruning and reduces reconstruction error through subsequent weight updates. Wanda (Sun et al., 2023) combines weight magnitudes with input activations to retain parameters that better align with the current data distribution. DSOT (Zhang et al., 2023c) proposes a parameter adjustment method to minimize the discrepancy between the source model parameters and the pruned model parameters. OWL (Yin et al., 2023) introduces non-uniform layered sparsity, which is advantageous for higher pruning rates.

2.2 Special Domain Fine-tune Model

Since the advent of the machine learning era, models have required adjustments on specific data to achieve desired performance. In the era of pre-trained language models, this approach has been slightly modified. Researchers first pre-train a general model and then fine-tune it on domain-specific data, with the primary goal of leveraging the capabilities of the pre-trained model. This is even more crucial in the era of large language models, resulting in the development of numerous models in different domains. For example, in the code domain (Rozière et al., 2023; Yu et al., 2023c; Luo et al., 2023b), mathematics domain (Luo et al., 2023a; Yue et al., 2023; Yu et al., 2023a; Gou et al., 2023; Yuan et al., 2023), medical domain (Kweon et al., 2023; Chen et al., 2023; Toma et al., 2023), and finance domain (Zhang et al., 2023a; Yang et al., 2023b; Xie et al., 2023).

Although we have obtained many fine-tuned models in specific domains, if we want a single model to have the capability to handle multiple domains, the fundamental approach is to fine-tune the model on all domain data together. However, this requires a significant amount of computational resources. Therefore, model fusion methods have gained attention.

2.3 Model Merge

The mainstream model fusion methods can be divided into four sub-domains: alignment (Li et al., 2016), model ensemble (Pathak et al., 2010), module connection (Freeman and Bruna, 2017), and weight averaging (Wang et al., 2020). Among these methods, only weight averaging reduces the number of model parameters, while the others require the coexistence of model parameters from multiple domains (Li et al., 2023b). Within the weight averaging sub-domain, there are also several approaches, such as subspace weight averaging (Li et al., 2023a), SWA(Izmailov et al., 2018), and task arithmetic (Ilharco et al., 2023). We are particularly interested in the task arithmetic sub-domain because it does not require the fusion of multiple models during the training process. Instead, it only requires obtaining the weights of a fully trained model.

The task arithmetic approach suggests that there is a domain-specific offset between the fine-tuned model weights and the base model weights. By adding or subtracting these offsets from multiple domains, it is possible to fuse or selectively exclude the capabilities of certain domains. Subsequent works have explored the application of task arithmetic to LORA (Zhang et al., 2023b; Chitale et al., 2023; Chronopoulou et al., 2023), as well as how to better fuse models and reduce conflicts between parameters. Ortiz-Jiménez et al. (2023) achieved this by scaling the coefficients of different models during the fusion process to mitigate conflicts between models. Yang et al. (2023a) further proposed adjusting the scaling coefficients at the model hierarchy level to address conflicts caused during model fusion at a finer granularity. Yadav et al. (2023) selected which model weights to retain at specific positions by comparing the absolute values of conflicting weights. Jin et al. (2023) adjusted the entire conflicting vector in vector space to ensure that the L2 distance between this vector and multiple original vectors remains equal.

2.4 Federated Learning

Federated learning is a setup where multiple clients collaborate to solve machine learning problems, coordinated by a central aggregator. This setup also allows for decentralized training data to ensure the privacy of data on each device (Zhang et al., 2021). Model fusion methods naturally possess the ability to combine locally trained models. Furthermore, since the central aggregator receives locally trained weights, there is no need to worry about data leakage issues.

3 Methodology

The purpose of our approach is to merge models from diverse domains into a single model with multi-domain capabilities. Therefore, we first review the definition of model merging.

Our approach consists of four parts, as shown in Fig. 1. First, we calculate the delta parameter, signifying the weight disparity between the fine-tuned models and the Base model. Second, we implement a variant of the magnitude pruning technique, referred to as DP, which discovers domain-specific significant parameters and removes redundant ones. This technique prunes the delta parameter to reduce parameter conflicts during model merging. Subsequently, we introduce a rescaling method, DPA, to amplify the significant parameters, resulting in enhanced performance. Conclusively, we merge the parameters from various fine-tuned models and incorporate them into the base model, thus yielding a single model with multi-domain capabilities.

3.1 Model Merging Problem

The purpose of model merging is to enhance the capability of a single model by combining models from multiple domains. 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.

3.2 Delta Parameter

For each model in each domain, we find the corresponding pre-trained model, known as the base model M^B. For domain i, we have the weights W^i of the model M^i and the weights W^B of the base model. We define the delta parameter as the transition of the parameter space distribution from the base model to the fine-tuned model, represented as $\Delta^i = W^B - W^i$. Analyzing the delta parameter enables a deeper understanding of the changes brought about by the fine-tuning process.

3.3 DPPA

First, we introduce Dynamically Pruning (DP) to discover domain-specific significant parameters and remove redundant ones. Subsequently, to enhance the capability in the domain, we propose Dynamical Partition Amplification (DPA), which amplifies significant parameters during the merging process.

3.3.1 DP: Dynamically Pruning

We propose using linear layers as the minimum unit and adjusting the pruning rate based on the significance of different linear layers. Here, the linear layers, such as Q, K, V, and O in Attention, and up/down sampling in MLP, are more fine-grained units compared to model layers. We first describe how to define the significance of parameters and then explain the method for adjusting the pruning rate.

Within the framework of OWL (Yin et al., 2023), the significance of a parameter is defined as the value exceeding the average weight magnitude by N-fold. We claim that this approach loses information when there is significant variation in the model parameters because it ignores the information about the magnitude of these parameters. Thus, we redefine significance. It now considers the accumulated magnitudes of parameters that are N times larger than the average magnitude. This improvement contains more comprehensive information about weight parameters. Based on empirical findings from OWL, we set N to 5. This approach allows us to determine the significance of parameters on both the model layer and the linear layer levels.

Once the significance of the parameters has been determined, we adjust the pruning rate accordingly. Following the principle that higher parameter significance corresponds to lower pruning rates, we define the pruning rate fluctuation at the model level as:

$$dif(\Delta_l) = -sig(\Delta_l) + \frac{1}{n}\sum_{l=1}^{n} sig(\Delta_l) \quad (1)$$

where dif represents the difference between significance and its mean. For simplicity, we reduce domain-specific Δ^i to Δ, thus Δ_l represents parameters in model layer l, $sig()$ represents the significance of the parameter, and n represents the number of model layers, respectively.

Furthermore, since the number of parameters in different linear layers may vary, we introduce a weighting factor for the parameter significance, as shown:

$$mean(\Delta_{lj}) = \frac{\sum_{l=1}^{n}\sum_{j=1}^{m} sig(\Delta_{lj}) * \|\Delta_{lj}\|_0}{\sum_{l=1}^{n}\sum_{j=1}^{m} \|\Delta_{lj}\|_0} \quad (2)$$

$$dif'(\Delta_{lj}) = -sig(\Delta_{lj}) + mean(\Delta_{lj}), \quad (3)$$

where Δ_{lj} represents parameters in model layer l linear layer j, m represents the number of linear layers in the model layer, $\|X\|_0$ represents the parameter count of X, respectively.

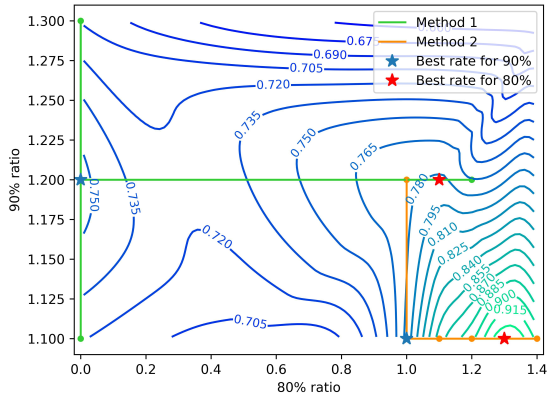

Figure 2: We utilize green and orange lines to represent the trajectories of the amplification rate search. Among them, the blue star represents the optimal rate searched at a 90% pruning parameter, while the red star represents the optimal rate searched at an 80% pruning parameter. The contour lines depict the specific performance in the mathematical domain.

Finally, we define the maximum value of pruning rate fluctuation, denoted as λ, based on previous experimental findings, and set it to 0.08. By considering both the fluctuation within the linear layer level and layer level, we derive the final pruning rate for each linear layer as follows:

$$norm(x) = \frac{x * \lambda}{max\ abs(x)} \tag{4}$$

$$\Theta_{lj} = \alpha + norm(dif(\Delta_l)) + norm(dif'(\Delta_{lj})), \tag{5}$$

where α represents original pruning rates, abs represents absolute value.

3.3.2 DPA: Dynamical Partition Amplification

After DP, we obtain the pruned delta parameters at various pruning rates. Our goal moving forward is to enhance performance while ensuring a consistent pruning rate. As the scaling rate increases, the model's performance shows a gradual decline after an initial rise. This pattern is consistently observed across various pruning rates, as illustrated in Fig. 2. Moreover, we postulate that during the fine-tuning stage, parameters with substantial deviations significantly influence the model's performance.

Therefore, we propose DPA, a method that dynamically modifies the enhancement factors for each division parameter at different pruning rates. We take into account two initialization methods to accomplish this dynamic adaptation and ultimately find the best outcomes. We select the initialization method with the best results as the final solution.

Method 1 We adjust the parameters in the 90% pruning rate partition by setting the rest to zero. The resulting curve of this method is illustrated by the green line in Fig. 2. We surmise that partitions with elevated pruning rates hold a greater degree of significance. Consequently, the precedence in sorting partitions is primarily influenced by their respective pruning rates. Illustratively, the parameters within the 90% pruning rate section are perceived as having a higher value compared to those within the 80% pruning rate partition. Upon the acquisition of the ideal amplification ratio, we progressively incorporate parameters from the 80% pruning rate partition, scaling only the newly included parameters.

Method 2 We employ the partition that aligns with the target pruning rate directly during the adjustment of the 90% partition. The resulting curve of this method is illustrated by the orange line in Fig. 2. We recognize that Method 1 generates excessively large amplification factors for more significant partitions, thereby causing a substantial displacement in the parameter space of partitions with lower pruning rates. This shift ultimately decreases performance when integrating parameters from partitions with lower pruning rates. In this strategy, when modifying more critical partitions, we consider the parameter distribution of less significant partitions. This method outperforms Method 1 when the pruning rate aim is high.

3.4 Model Merging with DPPA

After applying DPPA, we integrate parameters derived from distinct models. In Section 2.3, we refer to multiple existing methodologies for model fusion. However, our primary objective is to enhance the pruning technique. As such, we employ AdaMerging (Yang et al., 2023a), a state-of-the-art merging approach, to confirm the parameter integration following the pruning process. It is worth mentioning that models destined for merging via fine-tuning originate from an identical pre-trained model, as existing fusion techniques do not support the integration of heterogeneous models.

Thus, we get the final merging model:

$$W^m = W^B + \Sigma_{i=1}^{k} \text{DPPA}(\Delta^i) \tag{6}$$

4 Experiments

4.1 Experimental Setup

Pre-Trained Backbone and Fine-tune Models We have taken into consideration the need to fine-tune the same base model for different domains and the impact of the base model's performance. Therefore, we have decided to choose LLaMa 2(Touvron et al., 2023b) as the base model, instead of LLaMa(Touvron et al., 2023a), Mistral(Jiang et al., 2023), or other pre-trained models. For the three domains, mathematics, finance, and law, we have selected three models with good performance, namely Abel(Chern et al., 2023), Finance-chat, and Law-chat(Cheng et al., 2023).

Datasets For each domain, we have chosen two datasets. In the mathematics domain, we have selected GSM8k(Cobbe et al., 2021) and MATH(Hendrycks et al., 2021). We evaluate the models' performance using zero-shot accuracy and utilize the testing script provided by Abel(Chern et al., 2023). As for the finance domain, we have chosen FiQA_SA(Maia et al., 2018) and FPB(Malo et al., 2014). As for the law domain, we have chosen SCOTUS (Spaeth et al., 2020) and the UNFAIR_ToS (Lippi et al., 2019). Similarly, we evaluate the models' performance using zero-shot accuracy. Since AdaptLLM(Cheng et al., 2023) does not provide a testing script, we consider the multiple-choice question to be correct when the predicted sentence contains the correct choice.

Evaluation Metric To evaluate the correlation between the pruned and dense model, we formulated the Task-Ratio metric. Furthermore, to exhibit the model's generalization proficiency within each domain, we decided to use two datasets. We established the Domain-Ratio as a measure for gauging the specialized capability of the pruned model within a particular domain. The formula for Domain-Accuracy is as follows:

$$\text{Task-Ratio}_j = \frac{R(M_{pruned}, T_j)}{R(M_{dense}, T_j)} \tag{7}$$

$$\text{Domain-Ratio} = \sqrt[n]{\Pi_{j=1}^{n} \text{Task-Ratio}_j}, \tag{8}$$

where $R(M, T)$ represents the performance of model M on task T, M_{dense} refers to the fine-tuned model, M_{pruned} represents the pruned model, and T_j represents task j within the given domain, respectively. According to the formula, the Domain-Ratio of the dense model is 100%.

Implementation Details In our study, we employed the vLLM framework for reasoning. For the datasets GSM8k and MATH, we set the batch size to 32. As for the FiQA_SA, FPB, SCOTUS, and UNFAIR_ToS datasets, we set the batch size to 1. We utilized a greedy decoding approach with a temperature of 0. The maximum generation length for all tasks was set to 2048. Our experiments were conducted using the NVIDIA Tesla A100 GPU.

4.2 Baseline Method

We establish two methods without pruning, two methods of pruning-base, and one of randomly deleting and scaling as baseline. they are described below:

- **Model Soups** (Wortsman et al., 2022) calculate the average value by adding all model parameters.
- **LM-Cocktail** (Xiao et al., 2023) weighted the models from different domains and chose the optimal result.
- **Magnitude** (Han et al., 2015b) sorts weights based on their absolute values, keeping weights with larger absolute values and removing weights with smaller ones.
- **OWL** (Yin et al., 2023) building upon magnitude pruning, this method considers that parameter significance varies across different layers of the model.
- **DARE** (Yu et al., 2023b) suggests that after pruning, the sum of parameter values should remain the same. Therefore, it initially performs random pruning and then expands the remaining parameters based on the pruning rate to achieve the original sum of parameter values.

4.3 Main Result of DPPA

The results of the dense model and two methods without pruning are shown in Table 2. The results of the pruning methods are shown in Table 1. We compare the results of DPPA with two magnitude-based pruning methods, as well as compare the results of DARE. The experimental results show that our approach retains only 20% of the specific domain parameters, yet achieves comparable performance to other methods that retain 90% of the specific domain parameters. Due to space limitation, we place the completed experimental table

Sparse ratio	Magnitude	OWL	DARE	DPPA
Math-Dense				
10%	96.46	96.69	96.64	-
80%	80.12	77.11	87.41	**97.08**
90%	53.41	54.09	73.44	**86.85**
Fin-Dense				
10%	90.81	89.12	91.04	-
80%	71.04	74.92	84.01	**96.65**
90%	54.71	56.74	82.90	**92.11**
Law-Dense				
10%	95.74	110.74	116.02	-
80%	113.98	**124.97**	79.93	116.02
90%	84.35	**121.42**	69.33	110.55

Table 1: Domain-Ratio of different pruning methods at various pruning rates. Additional results under different pruning rates and the performance on a single dataset are presented in Appendix C.

Domains	Dense	Model Soups	LM-Cocktail
Math	100	15.99	76.96
Fin	100	79.46	78.80
Law	100	93.98	105.77

Table 2: Domain-Ratio of dense model and two methods without pruning.

in Appendix C. The comparison of the results of the two initialization methods in DPA is placed in Appendix A.

4.4 Abnormal Situations in Law Domain

We believe that our method can achieve performance levels as close as possible to the dense model itself. However, for some tasks that require performance beyond what the dense model can offer, our method may not be as effective. In contrast to the expected results from normal pruning, in the law domain, the pruned models significantly outperformed the dense model. The best performance was observed in the range of 120-140% of the dense model's performance, as pruning rates varied from 10% to 90%. We attribute this phenomenon to two factors: first, the relatively low performance of the law domain finetune model itself, and second, the possibility that the model was in a local minimum, causing any offset introduced by pruning to enhance the model's performance.

4.5 The Effectiveness of DP

As shown in Table 3, DP achieves better performance at high pruning rates. This is because DP adjusts the significance of linear layer parameters within each layer, allowing for the retention of more crucial parameters at high pruning rates.

Domains	Magnitude	OWL	DP
Math	53.41	54.09	**54.97**
Fin	54.71	56.74	**62.06**
Law	84.35	**121.42**	110.55

Table 3: Domain-Ratio of DP at a pruning rate of 90%.

Domains	DARE	DARE+DPA	DPPA
Math	73.44	83.63	**86.85**
Fin	82.90	85.08	**92.11**
Law	69.33	**120.89**	110.55

Table 4: Domain-Ratio of DARE using DPA at a pruning rate of 90%.

4.6 The Generality of DPA

We investigated the generality of the DPA method by applying it to the state-of-the-art method, DARE. Considering that the DARE method already amplifies the parameters and achieves significant amplification at high pruning rates (5 times for 80% and 10 times for 90%), we modified the approach to dynamic reduction instead. Following the methodology, we conducted experiments, and the results are presented in Table 4.

4.6.1 When can DP replace DARE?

According to the DARE paper, the method's performance is not satisfactory when the maximum float value of the deviation between the parameters and the base model exceeds 0.03. Our observations indicate that the larger the offset, the poorer the performance. This is evident from the parameter offset presented in Table 5. Certainly, we will present more comprehensive results in Appendix B. When DARE falls below 90% performance at a pruning rate of 90%, our method can serve as a viable alternative.

4.7 Why DPPA is Useful?

To investigate this question, we analyzed the Delta parameters, as shown in Fig 3. We explored the relationship between the remaining parameters after

Model	Min	10%	90%	Max
Math-Dense	-0.01733	-0.00114	0.00114	0.02014
Fin-Dense	-0.02612	-0.00160	0.00160	0.02011
Law-Dense	-0.02185	-0.00158	0.00158	0.02027

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

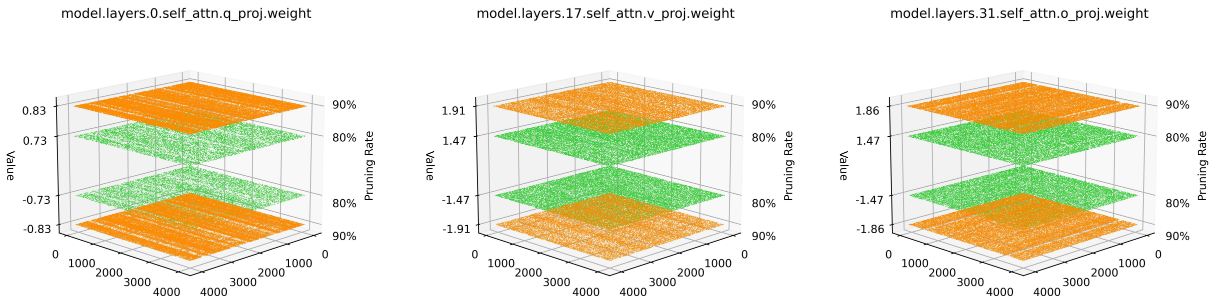

Figure 3: After analyzing the pruned parameters of the financial model, it is evident that there is a higher parameter count in the initial and final 0, 31 layers, while the middle 17 layers have fewer parameters. Additionally, in the Q, K, V components, it is observed that 90% of the parameters are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

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

Method & Pruning Rate	Math	Fin
DARE 90%	21.10	64.88
DPPA 90%	89.25	79.40
DARE 80%	58.43	77.16
DPPA 80%	**92.75**	**95.45**

Table 7: Domain-Ratio of the model that combines domains mathematics and finance.

DP at different pruning rates and different linear layers. The graph indicates that although DP is an unstructured pruning method, it exhibits some characteristics of structured pruning in the results of high pruning rates for the Delta parameters. This dimension partitioning provides some interpretability for the distribution of parameter space in specific domains. Therefore, when we use DPA, by amplifying the parameters, we strengthen the weights of the domain in these dimensions and restore certain capabilities.

4.8 Main Result of Merge Methods

We validate the effectiveness of our pruning method for the task of model fusion by integrating models. In Table 6, we present the merging results for three domains, while in Table 7, we showcase the merging results for two domains. We choose pruning rates of 80% and 90% to compare the results of model merging, as shown in the Table 7. Based on the results, our method demonstrates an improvement of nearly 20% in performance compared to DARE at the same pruning rate. This finding substantiates the efficacy of our pruning approach in the context of complex model fusion.

By comparing the results in Table 6 and Table 7, It can be observed that the integration of a fine-tuned model from an additional domain considerably influences the performance of DARE, causing significant performance deterioration. In comparison, our method achieves comparable performance. Upon augmenting an additional domain, there has been a decrease in performance in other domains at varying pruning rates. This outcome is consistent with expectations because parameter conflicts are a common issue with model merging, invariably resulting in performance degradation.

5 Conclusions

In this study, we introduce a pruning method called DP, which is an improved approach based on amplitude 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. Additionally, we investigate the underlying reasons behind the effectiveness of our proposed method.

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 fusion, there remains the issue of performance degradation.

References

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.

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. 2023a. Low dimensional trajectory hypothesis is true: Dnns can be trained in tiny subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(3):3411–3420.

Weishi Li, Yong Peng, Miao Zhang, Liang Ding, Han Hu, and Li Shen. 2023b. Deep model fusion: A survey. *CoRR*, abs/2309.15698.

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.

Marco Lippi, Przemyslaw Palka, Giuseppe Contissa, Francesca Lagioia, Hans-Wolfgang Micklitz, Giovanni Sartor, and Paolo Torroni. 2019. CLAUDETTE: an automated detector of potentially unfair clauses in online terms of service. *Artif. Intell. Law*, 27(2):117–139.

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.

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.

Harold J. Spaeth, Lee Epstein, Jeffrey A. Segal, Andrew D. Martin, Theodore J. Ruger, and Sara C. Benesh. 2020. Supreme court database, version 2020 release 01. Washington University Law.

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, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

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. 2023b. 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. 2023. Resolving interference when merging models. *CoRR*, abs/2306.01708.

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.

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%.

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 initialization methods

We show a performance comparison of the two initialization methods at 80% pruning rate in Table 8.

Model	Min	10%	20%	30%	40%	50%	60%	70%	80%	90%	Max
Math-Dense	-0.0173	-0.0011	-0.0007	-0.0004	-0.0002	1.175e-08	0.0002	0.0004	0.0007	0.0011	0.0201
Fin-Dense	-0.0261	-0.0016	-0.0010	-0.0006	-0.0003	0.0	0.0003	0.0006	0.0010	0.0016	0.0201
Law-Dense	-0.0218	-0.0015	-0.0010	-0.0006	-0.0003	0.0	0.0003	0.0006	0.0010	0.0015	0.0202

Table 9: The offset of different models from the base model at different position proportions.

B The Offset of Models

We presented ten different percentage values in Table 9.

C Main Result of Various Pruning Methods on Specific Tasks

We presented all pruning results in Table 10 and Table 11.

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	0.595744681	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

Table 10: All pruning result for three domain.

Sparse ratio	Magnitude	OWL	DP	DARE
FPB				
0.1	0.642268041	0.631958763	0.58556701	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
UNFAIR_ToS				
0.1	0.191860465	0.238372093	0.26744186	0.203488372
0.2	0.284883721	0.279069767	0.186046512	0.191860465
0.3	0.25	0.261627907	0.209302326	0.238372093
0.4	0.244186047	0.220930233	0.25	0.180232558
0.5	0.197674419	0.209302326	0.197674419	0.203488372
0.6	0.279069767	0.244186047	0.209302326	0.226744186
0.7	0.209302326	0.23255814	0.261627907	0.220930233
0.8	0.186046512	0.25	0.244186047	0.13372093
0.9	0.215116279	0.26744186	0.255813953	0.145348837
SCOTUS				
0.1	0.216666667	0.233333333	0.233333333	0.3
0.2	0.316666667	0.283333333	0.283333333	0.266666667
0.3	0.283333333	0.25	0.283333333	0.266666667
0.4	0.266666667	0.316666667	0.35	0.25
0.5	0.25	0.233333333	0.35	0.166666667
0.6	0.316666667	0.35	0.3	0.116666667
0.7	0.35	0.35	0.35	0.233333333
0.8	0.316666667	0.283333333	0.25	0.216666667
0.9	0.15	0.25	0.216666667	0.15

Table 11: All pruning result for three domain.