From Bottom to Top: Extending the Potential of Parameter Efficient Fine-Tuning

Jihao Gu, Zelin Wang, Yibo Zhang, Ziji Zhang, Ping Gong*
School of Artificial Intelligence,
Beijing University of Posts and Telecommunications, BeiJing, China
{gujihao, wang_zelin, zhangyibo, 1614823271, pgong}@bupt.edu.cn

Abstract

With the proliferation of large language models, Parameter Efficient Fine-Tuning (PEFT) method, which freeze pre-trained parameters and only fine-tune a few task-specific parameters, are playing an increasingly important role. However, previous work primarily applied uniform operations across all layers of the model, overlooking the fact that different layers in a transformer store different information. In the process of exploration, We find that there is a significant differences in fine-tuning strategies between different layers, and fine-tuning only a subset of layers can even achieve comparable performance. Based on this, we propose the Hybrid LoRA-Prefix Tuning (HLPT) method, which uses enhanced LoRA and Prefix-tuning methods with learnable adaptive mechanism separately for the bottom and top layers, and the Half Hybrid LoRA-Prefix Tuning (H^2LPT) method, which goes a step further, reducing the parameter count to nearly half by omitting fine-tuning in the middle layers. Extensive experiments with large language models on various downstream tasks provide strong evidence for the potential of PEFT focusing on different layers' interactions and the effectiveness of our methods. Furthermore, we validate the robustness of these methods and their advantages in smoothing training convergence, reducing inference time requirements.

1 Introduction

With the success of large language models like GPT-3 (Brown et al., 2020) and Ernie3.0 (Sun et al., 2021), large language models (LLMs) have demonstrated remarkable generative capabilities, enabling them to handle various downstream tasks. Moreover, it has spurred the development of numerous large language models such as GPT-J (Wang and Komatsuzaki, 2021) and LLaMA (Touvron et al., 2023) . However, this also brings forth a challenge that these LLMs typically have billions of parameters, making it challenging to afford the computational resources required for fine-tuning all parameters of the model for different downstream tasks.

To address this challenge, the PEFT (Houlsby et al., 2019; Li and Liang, 2021; Hu et al., 2021; He et al., 2022a) method, as an emerging fine-tuning approach, often adds a subset of parameters to the model to tune while freezing the original pre-trained model parameters. Significantly reducing the number of parameters needed for fine-tuning while achieving comparable results on downstream tasks, thus greatly reducing resource consumption during training.

In previous works, the different information storage between layers was neglected while most of works applied the same method uniformly across all layers of the model. However, prior researches have indicated the distinct effect of different layers. (Jawahar et al., 2019; Tenney et al., 2019; Jawahar et al., 2019). It is generally believed that lower layers contain more specific sentence-level information, middle layers focus on capturing syntactic information and higher layers store more abstract semantic information. There has been a lack of analysis and utilization concerning the distinct roles played by different layers. Based on the perspective, we assume that by effectively combining various PEFT methods according to the distinct roles of different layers, we can achieve a more efficient PEFT approach, and conduct extensive experiments using LLMs with zero-shot learning to validate our ideas and the increasingly important generation capability of our methods in contrast to previous methods focusing on understanding tasks.

Inspired by the work (Patel et al., 2021), we focuses on two efficient methods: LoRA, which operates in parallel mechanisms and utilizes matrix rank, and prefix, which adds virtual heads before multi-head attention. As shown in Figure1, in our

*Ping Gong is the corresponding author.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 3488–3500

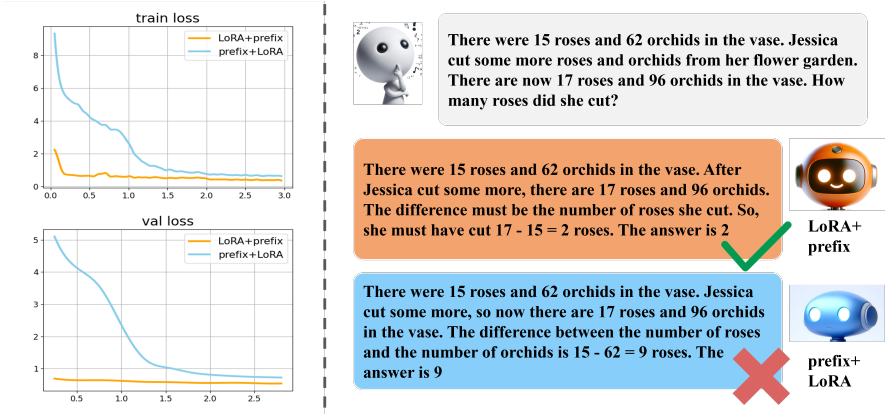

Figure 1: This is an analysis of the experimental results regarding the effects of different layers. We divide the LLaMA-7B model into two equal parts based on the number of layers, where "LoRA+prefix" represents placing LoRA at the bottom layers and prefix at the top layers, and "prefix+LoRA" represents the opposite. On the left side is a comparison of the loss during convergence for the two training methods, with the x-axis representing the number of epochs. On the right side is a comparison of the responses generated by the two trained models for the same question.

study, we find different layer allocation strategies for LoRA and prefix methods result in completely opposite effects on task performance and model convergence speed. We attribute this result to that LoRA fundamentally adjusts the model parameters, making it suitable for fine-tuning general specific information, whereas prefix tuning involves adding prompts to the input to guide the model to adapt to downstream tasks, requiring a certain level of understanding of large model language, making it suitable for capturing abstract information at the top layers. Further exploration (refer to Section 3) has revealed that the model can even maintain comparable performance while discarding half of the layers during fine-tuning.

Based on the survey mentioned above, we have introduced two methods: Hybrid LoRA-Prefix Tuning (HLPT), which effectively combines the enhanced LoRA and Prefix methods operating on both the top and bottom layers, and Half Hybrid LoRA-Prefix Tuning (H^2LPT), which further reduces the number of parameters by half based on HLPT excluding the operations on the middle layers, to help the model adapt more effectively to various downstream tasks and unleash the potential of PEFT. We conduct lots of experiments on a total of six datasets for mathematical reasoning tasks using large language models, namely LLaMA-7B, LLaMA-13B, and GPT-J (6B) (Touvron et al., 2023; Wang and Komatsuzaki, 2021). The accuracy can be even significantly improved while reducing the number of parameters by up to 71 points compared to the original LoRA and Prefix methods in the main experiment. These experiments highlight the powerful potential of tuning each layer differently based on interactions between different layers to reduce model resource consumption and enhance performance, affirming the effectiveness of the methods presented in this paper.

In summary, our main contributions can be outlined as follows:

1.Investigate and demonstrate the suitability of distinct information storage in different layers for fine-tuning methods.

2.Introduce two methods, HLPT and H^2LPT, employing enhanced LoRA and Prefix to adapt to the diverse information storage across layers.

3.Discover the conclusion that the model can ignore half of the layers during fine-tuning while still achieving comparable performance, and propose a corresponding approach.

2 Background and Related Works

As the model size increases, the significance of PEFT methods becomes increasingly apparent, leading to the development of several classical and widely-used PEFT techniques. These methods predominantly involve freezing the original pre-trained parameters and fine-tuning only the newly introduced parameters. Among these, "Adapter tuning" adds adjustable modules to the multi-head attention and feed-forward layers of the transformer model. "Prefix tuning" (Li and Liang, 2021) intro-

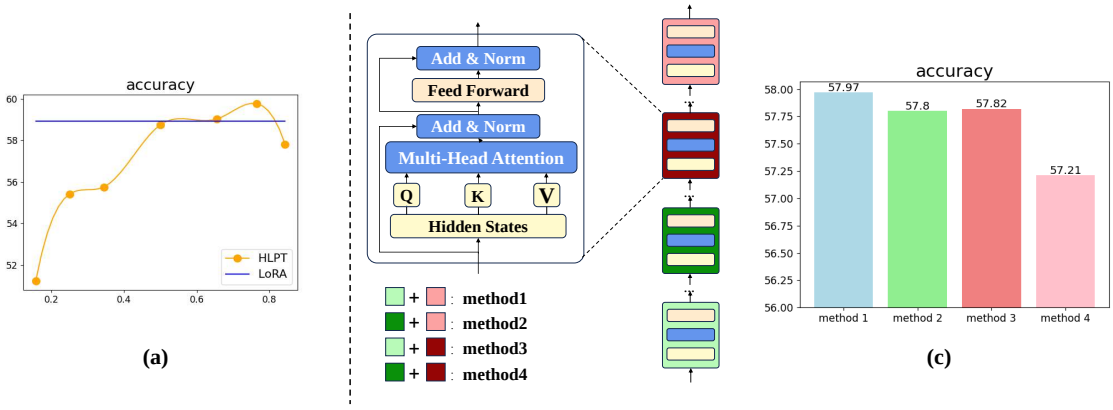

Figure 2: Figure(a) is a line graph depicting the average accuracy across six datasets as a function of the ratio of lower-level LoRA to upper-level Prefix layers. The orange line represents the corresponding ratio for the hybrid method, while the blue line represents the LoRA method. The seven points on the orange line correspond to the experimental settings for the LLaMA-7B model, which has 32 transformer layers, with different ratios of LoRA and Prefix layers: (5:27), (8:24), (11:21), (16:16), (21:11), (24:8), and (27:5), totaling 7 sets of experiments.The x-axis represents the ratio of layers between LoRA and Prefix, and the y-axis represents the average accuracy of the LLaMA-7B model.Figure(b) is an explanatory figure of fine-tuning only specific layers, the layers fine-tuned by each of the four fine-tuning methods are color-coded, green represents the layers fine-tuned with the LoRA at the bottom, while red represents the layers fine-tuned with the prefix at the top in the chosen method, and figure(c) is a comparison of the average experimental results on mathematical reasoning datasets for the LLaMA-7B model using the four methods mentioned above.

duces learnable parameter vectors in front of the input "k" and "v" of each multi-head attention layer in the transformer, expanding the dimensions of "k" and "v." "LoRA" (Hu et al., 2021) incorporates learnable parallel models alongside the multi-head attention layers.

On the other hand, many methods have attempted to reduce the number of transformer layers in the model (Xin et al., 2020; Fan et al., 2019). Subsequently, the AdapterDrop (Rücklé et al., 2021) method was introduced, which builds upon the PEFT approach and investigates the impact of removing fine-tuning in the lower layers of the model on both speed and performance. In addition, there are also some studies exploring the prompt tuning methods and the function of neurons in different layers of pretrained transformer models (Su et al., 2021; Wang et al., 2022).

Building on the excellent PEFT approach mentioned above, many studies have attempted to integrate existing methods together. The "UniPELT" (Mao et al., 2022) method combines all these methods into a single transformer model by introducing different gating mechanisms for each model, achieving a unified model overall. The "Sparse" (Hu et al., 2022) explore constructing delta-tuning modules in an automatic manner. "Unified view" (Patel et al., 2021) employs mathematical analysis to demonstrate the commonalities among adapter, LoRA, and Prefix models using a set of similar mathematical formulas. It finds that multi-head attention achieves better results with fewer parameters, parallel structures outperform serial structures, and adding parallel models to the feed-forward layers yields superior results. Ultimately, it proposes a Prefix-tuning with parallel adapter structure across the entire model, highlighting the effectiveness of multi-head attention's Prefix and parallel LoRA in the feed-forward layers. This also guides the use of LoRA and Prefix in this paper to adapt to the different roles of various layers.

Compared to these methods above, We believe our contribution lies in exploring the interactions between different layers aligned with PEFT methods, clearly delineating the HLPT method and the H^2LPT method that omits intermediate layer fine-tuning. Our method can improve the effectiveness as reduce resource consumption. At the same time, we primarily verify the increasingly important generation capability of the method in the era of LLMs.

3 Analysis of layers

To further explore the differences in the combination of the two methods across different layers

following the findings in figure 1, We conduct experimental analysis in this chapter.

3.1 Analysis of Different Layer Allocation Methods

We conduct a comparative exploration on LLaMA-7B using only Prefix and LoRA (LoRA is applied to the feed-forward network layers to further widen its gap with prefix applied to the attention layer, adapting to different inter-layer information), with different allocation methods based on the structure where LoRA is at the bottom layers and prefix is at the top layers, the further reasons for using these two methods can be found in the appendix E. As shown in Figure 2(a), it can be observed that as the proportion of LoRA in the lower layers increases, the model's performance initially improves and then declines. The best performance is achieved when LoRA was applied to 24 layers and Prefix to 8 layers. It is determined that a 3:1 ratio of LoRA in the lower layers and Prefix in the upper layers achieves better performance. Based on this, the final model proposed in this paper also demonstrated superior performance on GPT-J and LLaMA-13B, highlighting the rationality and universality of this allocation method.

3.2 Analysis of fine-tuning only a subset of layers

During our research, we find that fine-tuning a subset of layers does not affect the performance. To further explore this phenomenon, we boldly propose the idea of fine-tuning half of the layers and conducte experiments. Based the previously chosen best method in Section 3.1, we propose four different parameter reduction methods, as illustrated in Figure 2(b). The final experimental results, as shown in Figure 2(c), indicate that the accuracy of all four methods does not exhibit a significant decrease compared to the best accuracy (59.78) in figure 2(a). We believe this is due to redundant information stored across layers, which allows for a significant reduction in the number of trainable parameters. Besides, among them, the method placing LoRA at the bottom layer and prefix at the top layer achieves the best results in our experiments.

4 Method

4.1 Motivation and Methods

Previous research has introduced several outstanding methods, but most of them either treat all layers of the model uniformly (He et al., 2022b; Mao et al., 2022; He et al., 2022a), or directly reduce certain layers of the model (Rücklé et al., 2021). We believe that by effectively utilizing various methods based on the varying storage of information across different layers of the model, we can further push the performance boundaries of the PEFT method.

According to the conclusion that different layers of the model store different information, we discover that different PEFT methods exhibit vastly different performance across layers. Meanwhile, based on the research in the paper (Patel et al., 2021) and our perspective that LoRA, which modifies the model's original parameters, is more suitable for capturing specific information at the lower layers, while prefix, which adds virtual tokens before input, is more suitable for capturing abstract information at the higher layers. We believe using LoRA and Prefix methods separately to adapt to the information stored in different layers can be explored to develop PEFT methods that are both resource-efficient and effective.

Based on the survey conducted in section 3, we have introduced two methods: Hybrid LoRA-Prefix Tuning (HLPT), and Half Hybrid LoRA-Prefix Tuning (H^2LPT), as depicted in Figure 3, our methods utilize LoRA added to the majority of lower layers in the model (LoRA is applied to the feed-forward network layers), while Prefix is applied in a smaller portion of the upper layers. A learnable gating unit is introduced for all models. This unit generates a (0,1) parameter based on the input of the attention layer and the feed-forward neural network for Prefix and LoRA separately through a Linear Layer, and this parameter is multiplied with Prefix parameters and the scaling of LoRA, resulting in adjusted parameters to help the model adapt more effectively to various downstream tasks and unleash the potential of PEFT.

To be specific, according to the performance shown in figure 2, in the experiments of this paper, HLPT involves using an improved LoRA in the bottom 3/4 of the model's layers and an enhanced Prefix method in the remaining 1/4 of the layers. H^2LPT is a further refinement, a method that only fine-tunes the bottom 3/8 (3/7 for GPT-J) and the top 1/8 (1/7 for GPT-J) of the layers. The experiments in Section 5.2 and Appendix D further demonstrate our methods and the general applicability of this allocation approach.

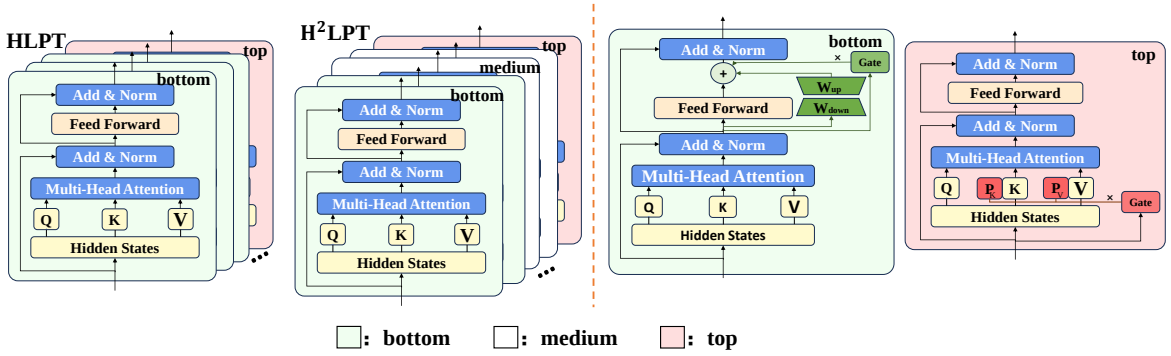

Figure 3: This is an illustration of the proposed two method in this paper. In the diagram, green and pink represent operations on the bottom and top Transformer layers, while white indicates middle layers without any operations. On the left are the two methods proposed in this paper, and on the right, specific operations are illustrated.

4.2 Details

The attention and FFN components of the transformer serve as the foundation of the method proposed in this paper, and the computations for these two components of the Transformer model are as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V \quad (1)$$

$$FFN(x) = ReLU(xW_1 + b_1)W_2 + b_2 \quad (2)$$

where $Q = xW_Q$ .The formulas for K and V follow the same logic. The LLaMA model used in this paper is further improved, and the computation of the feed-forward neural network layer is as follows:

$$FFN_{LLaMA}(x) = (Swish_1(xW_1) \otimes xV)W_2 \quad (3)$$

4.3 Prefix Tuning at Top

Prefix tuning introduces virtual tokens of length "l" in the Transformer model. These tokens are used to concatenate with the original input "keys" and "values" parameters of the multi-head attention. Specifically, virtual tokens $P_k \in R^{l \times d_{hidden}}$ and $P_v \in R^{l \times d_{hidden}}$ are generated randomly and added in front of the input parameters Q and V. Therefore, the computation in the Transformer becomes as follows:

$$Attention(Q, K', V') = softmax(\frac{QK'^T}{\sqrt{d}})V' \quad (4)$$

where $K' = [P_k : K]$,$V' = [P_v : V]$. During fine-tuning for adaptation to downstream tasks, the model adjusts only P_k and P_v.

To enable the model to adapt to various downstream tasks, a gating unit is introduced. This unit takes the input x of the model at that layer and passes it through an Linear Layer to obtain an output $g_p \in (0, 1)$. This output is then multiplied with the P_k and P_v, and the new virtual token is added at the front of the model as follows:

$$P'_k = g_p P_k; P'_v = g_p P_v \quad (5)$$

where $g_p = Linear(x)$

4.4 LoRA at Bottom

LoRA introduces parallel low-rank parameters to the multi-head attention mechanism. Specifically, two parameters, $W_{down} \in R^{d_{hidden} \times r}$ and $W_{up} \in R^{r \times d_{hidden}}$, are added alongside the query and value attention. The original multi-head parameter is W_q, and for the calculation of Q, the formula is as follows:

$$Q = x(W_Q + \alpha W_{down} W_{up}) \quad (6)$$

Where α is a fixed parameter used to adapt to different downstream tasks. In this work, drawing from the previous work (He et al., 2022a), LoRA is applied to W_2 of the feed-forward neural network (FFN) with the following calculation formula:

$$FFN(x) = ReLU(xW_1 + b_1)(W_2 + \alpha W_{down} W_{up}) + b_2 \quad (7)$$

For the LLaMA model, LoRA is also applied to W_2 of the FFN, with the following formula.

$$FFN_{LLaMA}(x) = (Swish_1(xW_1) \otimes xV)(W_2 + \alpha W_{down} W_{up}) \quad (8)$$

Just like Prefix tuning, LoRA also incorporates a gate mechanism. Here, $g_l \in (0, 1)$ is computed using the input of the feed-forward neural network h_{FN}. The output is then multiplied with the fixed α. The new learnable α' is as follows:

$$\alpha' = g_l \alpha \tag{9}$$

where $g_l = Linear(h_{FN})$

5 Experiment

5.1 Setup

Followed by prior work (Hu et al., 2023),We conduct experiments on six math reasoning datasets using large language models, GPT-J-6B (Wang and Komatsuzaki, 2021) and LLaMA-7B/LLaMA-13B (Touvron et al., 2023), with pre-trained parameters from Hugging Face (Wolf et al., 2020).The datasets are (1) the SVAMP (Patel et al., 2021), (2) the AQuA (Ling et al., 2017), (3) the AddSub (Hosseini et al., 2014) dataset, (4) the MultiArith (Roy and Roth, 2016) dataset, (5) the SingleEQ (Koncel-Kedziorski et al., 2015) dataset, and (6) the GSM8K (Cobbe et al., 2021).

5.2 Main results

In Table 1, we present our main results, comparing our proposed two methods, HLPT and H^{2}LPT, with the original LoRA and Prefix-tuning methods on three models. (As any improvements made to the classical LoRA and prefix methods can be incorporated into our method with the expectation of achieving better results.) In all four methods, HLPT has a parameter count that is similar to the baseline, while H^{2}LPT has approximately half the parameter count. Furthermore, we also experiment with the parallel adapter proposed in the paper (He et al., 2022a) that inspired our work, and this method achieves slightly inferior results compared to our approach, despite having an order of magnitude larger number of parameters compared to our methods. For LLaMA-7B, our two proposed methods show an increase of up to 35% compared to the baseline. For LLaMA-13B and GPT-J, there is an increase of up to 20% and 8%, respectively.

It's worth noting that, the method of reducing training parameters by almost half, H^{2}LPT, performs better when using LLaMA-13B. This demonstrates the potential of this method, and also leads us to notice that not all layers need fine-tuning; there may be redundancy in inter-layer information storage, and fine-tuning only a subset of layers can even yield better results with limited source. Additionally, to demonstrate the robustness and versatility of our experiments, we split the original dataset of almost 3k data into 1k and 0.5k data using seed=42 separately, and conducte experiments using the LLaMA model(the results of GPT-J model can be seen in Appendix D). The results, as shown in Table 2, reveal that the baseline method significantly declines as the dataset size decreases (the Prefix method even struggle to complete tasks and generate fluent conversations with less data), while our proposed methods remain relatively stable and obviously outperform the baseline.

The hybrid methods used in this paper can make the model converge more smoothly, as shown in Figure 4. On both the training and validation datasets, the HLPT method proposed in this paper consistently exhibits a significantly better convergence trend compared to the baseline.

5.3 Ablation

To compare the impact of various modifications on model performance, we conduct ablation experiments, including experiments on placing LoRA on the attention layer and whether to include gates. As shown in Table 3, we find removing any of these methods would affect the model's performance to some extent, indicating the effectiveness of both of methods. Among these, gate serves as a supplementary function, aiding in adjustments based on different datasets for various models, and placing LoRA on the FFN amplifies the gap between it and the Prefix method, further highlighting the effectiveness of our proposed approach for rational fine-tuning of layers based on their interplay.

5.4 Discussion of Parameters

To validate the effectiveness of our methods in reducing training parameters while maintaining performance, we conduct experiments to compare them with LoRA, which shows better results in our experiment, using LLaMA-7B. We reduce LoRA's hyperparameter r to 4, halving the training parameters to compare with the H^{2}LPT. Furthermore, we compare both methods in extreme cases by using our proposed HLPT method and reducing the number of fine-tuned layers to 4 (3 at bottom and 1 at top) and decreasing LoRA's r to 1. In both cases, we reduce the parameter count to 1/8 of the original, and as shown in Table 4, our methods demonstrate the best performance, confirming their

Method	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
LLaMA-7B							
Prefix	42.50	23.53	58.23	60.00	66.67	15.91	44.47
LoRA	56.00	19.61	77.22	90.00	**87.25**	23.48	58.93
Parallel	63.00	**25.49**	75.95	86.36	83.33	23.11	59.54
HLPT	**60.50**	17.65	**78.48**	**90.91**	83.33	**28.79**	**59.94**
H^2LPT	60.00	15.69	73.42	86.36	84.31	**28.79**	58.09
LLaMA-13B							
Prefix	58.00	29.41	72.15	78.18	82.35	22.73	57.14
LoRA	67.50	29.41	86.08	92.73	89.22	34.09	66.50
Parallel	69.00	17.65	81.01	93.64	86.27	27.27	62.47
HLPT	**73.50**	**31.37**	84.81	90.91	**91.18**	**37.88**	68.27
H^2LPT	69.50	**31.37**	**88.61**	**94.55**	90.20	35.61	**68.30**
GPT-J-6B							
Prefix	41.50	9.80	67.09	75.45	71.57	9.85	45.88
LoRA	47.50	7.84	60.76	70.00	**76.47**	10.61	45.53
Parallel	42.50	**19.61**	56.96	78.18	66.67	12.88	46.13
HLPT	49.00	13.73	67.09	**82.73**	70.59	**15.15**	**49.71**
H^2LPT	**50.50**	17.65	**69.62**	73.64	68.63	14.39	49.07

Table 1: The results on six different mathematical reasoning datasets. The answer is the accuracy of calculations obtained using the zero-shot learning method on LLaMA-7B, LLaMA-13B, and GPT-J presented in the table.(HLPT: Hybrid LoRA-Prefix Tuning; H^2LPT:Half Hybrid LoRA-Prefix Tuning; **bold**: the best score; underline: the second best)

Method	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
LLaMA-7B(1000)							
Prefix	23.00	25.49	48.10	21.82	58.82	4.55	30.30
LoRA	44.50	23.53	**75.95**	81.82	**82.35**	**21.59**	54.96
HLPT	53.00	17.65	73.42	**86.36**	78.43	21.21	55.01
H^2LPT	**54.00**	**27.45**	73.42	82.73	79.41	20.08	**56.18**
LLaMA-7B(500)							
Prefix	0.00	0.00	1.96	0.00	0.00	0.00	0.33
LoRA	43.00	15.69	70.89	**58.18**	**78.43**	14.39	46.76
HLPT	45.00	15.69	**72.15**	55.45	69.61	14.39	45.38
H^2LPT	**48.00**	**23.53**	67.09	51.82	74.51	**15.91**	**46.80**
LLaMA-13B(1000)							
Prefix	41.00	23.53	60.76	65.45	63.73	14.02	44.75
LoRA	67.00	**27.45**	78.48	82.73	**90.20**	28.41	62.38
HLPT	**69.50**	23.53	**86.08**	89.09	**90.20**	33.33	**65.28**
H^2LPT	65.00	19.61	78.48	**93.64**	**90.20**	**34.09**	63.50
LLaMA-13B(500)							
Prefix	15.00	21.57	32.91	7.27	43.14	4.55	20.74
LoRA	57.50	25.49	74.68	**74.55**	84.31	**28.03**	57.43
HLPT	**59.50**	**27.45**	79.75	73.64	**87.25**	22.35	**58.32**
H^2LPT	58.50	19.61	**86.08**	70.00	86.27	25.00	57.58

Table 2: The experimental results of the model using the LLaMA model with dataset sizes of 1k and 0.5k.(**bold**: the best score; underline: the second best)

Setting	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
HLPT	60.50	17.65	**78.48**	**90.91**	**83.33**	**28.79**	**59.94**
w/o gate	**62.50**	**25.49**	**78.48**	**90.91**	77.45	23.86	59.78
on atten	59.00	**25.49**	**78.48**	88.18	**83.33**	21.21	59.28
H^2LPT	**60.00**	15.69	73.42	**86.36**	**84.31**	**28.79**	**58.09**
w/o gate	58.50	**23.53**	**75.95**	85.45	**84.31**	20.08	57.97
on atten	57.50	19.61	73.42	**86.36**	81.37	21.21	56.58

Table 3: Ablation experiments conducted on LLaMA-7B for HLPT and H^2LPT. These experiments involve removing the gate and parallelizing LoRA back to attention layers(on atten), respectively.(**bold**: the best score)

Setting	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
H^2LPT	**60.00**	15.69	73.42	**86.36**	84.31	**28.79**	**58.09**
LoRA-	52.00	**23.53**	**74.68**	**86.36**	**86.27**	24.24	57.85
HLPT*-	**47.50**	**27.45**	72.15	73.64	**83.33**	19.32	**53.90**
LoRA*-	46.00	21.57	67.09	**77.27**	82.35	**20.83**	52.52

Table 4: Comparison between H^2LPT and LoRA-, HLPT– and LoRA– on LLaMA-7B. (LoRA-:LoRA with r=4; HLPT*-: HLPT with four layers to fine-tune; LoRA*-: LoRA with r=1).

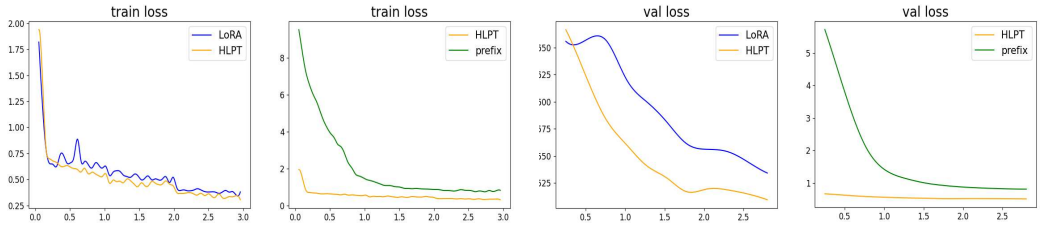

Figure 4: The comparison of HLPT method(orange) in terms of smoothing the optimization process relative to LoRA(blue) and Prefix(green) methods. The x-axis represents the training epochs, and the y-axis represents the loss on the training and validation datasets. All experiments are conducted on the LLaMA-7B model.

effectiveness.

Setting	Params	Time	Avg.
LLaMA-7B			
Prefix	7.86M(100.0%)	100.0%	100.0%
LoRA	4.19M(53.33%)	71.0%	181.4%
HLPT	4.51M(57.32%)	71.7%	181.6%
H^2LPT	2.25M(28.66%)	65.5%	185.4%
LLaMA-13B			
Prefix	12.29M(100.0%)	100.0%	100.0%
LoRA	6.55M(53.3%)	85.1%	139.4%
HLPT	7.06M(57.5%)	84.2%	145.9%
H^2LPT	3.53M(28.7%)	80.9%	141.9%

Table 5: Comparison in terms of parameters, time, and performance. Experiments were conducted on the LLaMA model trained with 1000 samples. 'Params' refers to trainable parameters, 'Times' indicates the total time the model was tested on six datasets, and all percentages are results obtained with Prefix as the baseline method.

In order to reveal the impact of reducing parameter count by nearly half in this paper, we conduct experiments using the LLaMA model with 1k training samples, as shown in Table 5. Our methods can even improve accuracy by 85.4% while reducing parameters by 71.3%. All of the above experiments confirm the value of this research.

6 Conclusion

In this paper, we have explored different layer-specific fine-tuning methods corresponding to the varying information stored at different layers of the model and the function of fine-tuning only a subset of layers, further pushing the boundaries of the PEFT approach. We conducted extensive experiments to analyze various allocation methods. In the process, we introduced the HLPT method, which involves improved LoRA in the majority of lower layers and enhanced Prefix in the minority of top layers. Building upon this, we propose the

H^2LPT method, which omits fine-tuning in the intermediate layers. A large number of experiments have all demonstrated the effectiveness and robustness of our proposed methods on LLMs. These methods can even achieve significantly better performance in terms of generation capability than the baseline with fewer parameters and inference time while they can smooth training convergence. All experiments highlight the enormous potential of utilizing different fine-tuning methods for layers with varying information storage.

7 limitation

While a significant number of experiments have demonstrated the effectiveness of our methods and justified the rationality of using different fine-tuning approaches for different layers, our proposed HLPT and H^2LPT methods still have some limitations. Firstly, we conduct experiments only on large language models with a decoder-only structure. However, this is not a significant limitation given that decoder-only structures are commonly used in large language models today. Secondly, for the H^2LPT method, although it has been shown to achieve better performance, this approach still has many areas worth exploring. There is still significant room for development in fine-tuning based on partial layers, and it may even be possible to explore layer allocation methods based on training-time gradient backpropagation to further reduce resource consumption during training. While our experimental results in this paper explore the interplay between layers during fine-tuning and our methods have demonstrated promising benefits, there are still more experiments worth further exploration to further fully substantiate our conclusion due to the limited resources in our experiment.

In the future, we will endeavor to conduct more experiments with additional datasets and models, make improvements to these methods, and further explore, in conjunction with theory, methods that align with the differential information storage across layers.

8 Ethic statement

The primary aim of our paper is to investigate and enhance the performance and efficiency of the PEFT method, making fine-tuning of state-of-the-art models for downstream tasks more accessible. We believe that the methods and open-source code in this work do not pose any potential risk. Additionally, all our experiments are conducted using open-source datasets and models.

References

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022a. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Shwai He, Liang Ding, Daize Dong, Jeremy Zhang, and Dacheng Tao. 2022b. SparseAdapter: An easy approach for improving the parameter-efficiency of adapters. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2184–2190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

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

Shengding Hu, Zhen Zhang, Ning Ding, Yadao Wang, Yasheng Wang, Zhiyuan Liu, and Maosong Sun. 2022. Sparse structure search for delta tuning. *Advances in Neural Information Processing Systems*, 35:9853–9865.

Zhiqiang Hu, Yihuai Lan, Lei Wang, Wanyu Xu, Ee-Peng Lim, Roy Ka-Wei Lee, Lidong Bing, and Soujanya Poria. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Scott Yih, and Madian Khabsa. 2022. UniPELT: A unified framework for parameter-efficient language model tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6253–6264, Dublin, Ireland. Association for Computational Linguistics.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Subhro Roy and Dan Roth. 2016. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Yusheng Su, Chi-Min Chan, Jiali Cheng, Yujia Qin, Yankai Lin, Shengding Hu, Zonghan Yang, Ning Ding, Xingzhi Sun, Guotong Xie, et al. 2023. Exploring the impact of model scaling on parameter-efficient tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15062–15078.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, et al. 2021. On transferability of prompt tuning for natural language processing. *arXiv preprint arXiv:2111.06719*.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan

Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

Xiaozhi Wang, Kaiyue Wen, Zhengyan Zhang, Lei Hou, Zhiyuan Liu, and Juanzi Li. 2022. Finding skill neurons in pre-trained transformer-based language models. *arXiv preprint arXiv:2211.07349*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online. Association for Computational Linguistics.

A Scientific Artifacts

The datasets we use include the mathematical reasoning dataset SVAMP (Patel et al., 2021), AQuA (Ling et al., 2017), AddSub (Hosseini et al., 2014), MultiArith (Roy and Roth, 2016), the SingleEQ (Koncel-Kedziorski et al., 2015), GSM8K (Cobbe et al., 2021), and the commonsense inference dataset ARC (Clark et al., 2018), Boolq (Clark et al., 2019), WinoGrande (Sakaguchi et al., 2021), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), and OBQA (Mihaylov et al., 2018). The pre-trained models we utilize are LLaMA-7B/13B (Touvron et al., 2023), and GPT-J-6B (Wang and Komatsuzaki, 2021). All the aforementioned datasets and models are open-source, and our work is solely for scientific research purposes, aligning with their original intent.

B Reproducibility Statement

Data Usage: The datasets and hyperparameter settings in this paper are mainly referenced from the prior work (Hu et al., 2023)'s open-source code. For the mathematical reasoning tasks, due to resource limitations, we choose the version with the smallest dataset in this work (Hu et al., 2023), and this dataset utilized a subset of the data from some of the datasets mentioned above in Appendix A. All six datasets are combined by randomly selecting 80% of each, resulting in a total of 3260 data points for training. Testing is then performed on the remaining data for each dataset. For commonsense inference tasks, considering the resource, 15k version of this work (Hu et al., 2023) are used for training, and testing is conducted on the seven datasets mentioned above. During training and testing, a prompt is added to the data: 'Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.'

Hyperparameter Settings: During training, we largely follow the prior work(Hu et al., 2023)'s open-source code in regard to hyper-parameters. The 'r' value for LoRA is set to 8, and the Prefix added token length 'l' is 20 in our hybrid methods. In this setup, the parameter quantities of LoRA and Prefix methods at each layer are very close, enabling a fair comparison of their performance at different layers. Bedides, the learning rate for our hybrid methods to 3e-3, and the regularization parameter α in Eq 5 of the main text is set to 2. For the prefix method, the learning rate is set to 3e-2, and the length of virtual tokens is set to 30. The learning rate for original LoRA method is set to 3e-4, 'r' value for LoRA is set to 8. The bottleneck size for the parallel adapter method is set to 256, and the learning rate is set to 3e-4. For the sake of fair comparison, the remaining hyprerparameters are set the same for each model and method. At the same time, to ensure comparability and reproducibility of experiments, the random seed for all experiments is set to 42 and all training epochs are set to 3. In my methods, the $linear$ part of our gating unit are initialized using the Kaiming uniform approach.

Model Usage: In this paper, we utilize three large models: LLaMA-7B/13B (Touvron et al., 2023), and GPTJ-6B (Wang and Komatsuzaki, 2021). All training and testing experiments are conducted using either a single Nvidia A40 or Nvidia

RTX4090.

C Background of Large Language Models

LLMs trained on extensive text corpora have demonstrated their ability to perform new tasks from textual instructions or a few examples(Brown et al., 2020). Many new open-source large language models trained on open datasets have started to emerge such as LLaMA(Touvron et al., 2023) and GPT-J(Wang and Komatsuzaki, 2021). These models contribute to democratizing research on LLMs, as they can be run on a single GPU. With the widespread adoption of these large language models, a trend to utilize zero-shot leaning and model generation capabilities to accomplish various downstream tasks has emerged and the model's generation capability becomes increasingly important. These new LLMs can even outperform larger models on downstream tasks.(Hoffmann et al., 2022)

D Supplementary Experiments

Due to space constraints in the main text, we present the complete comparative experiments here. As shown in Table 6, we obtain consistent results with the LLaMA model in Table 2 using the GPT-J-6B model. Our method maintains stable performance even with a significant reduction in training data and outperforms the baseline methods. At the same time, with a dataset size of 1000, H^2LPT can even achieve the best average results, further highlighting the superiority of our method of fine-tuning by ignoring half of the layers.

Despite the findings of this paper (Su et al., 2023) indicating a reduction in differences among various PEFT methods as model size increases, the differences between them cannot be ignored. As demonstrated by the experiments in this paper (Hu et al., 2023), even with the LLaMA-13B model, there are still noticeable discrepancies among different methods. Additionally, we conduct extended experiments using GPT-J-6B on commonsense reasoning datasets (the learning rate for the H^2LPT method on this datasets was set to 5e-3. The other settings remain unchanged.), and the results showed on table 7 further validate the effectiveness, generalizability, and value of our methods.

We add the detaile results of the experiments shown in Figure 2 of the main text, where LoRA and Prefix occupy different proportions, across each six dataset. The experimental results are presented in Table 8.

We also supplement the specific accuracy for each dataset from the experiment in Figure 2, where half of the layers were omitted, and conduct further extended experiments. The results can be found in table 9. In the extended experiments using LLaMA-7B, 75% of the layers are omitted, we only using LoRA in the bottom 6 layers and Prefix in the top 2 layers. It can be found that the accuracy drop across six mathematical reasoning datasets was minimal, demonstrating the value of this layer-omission fine-tuning method.

E Further Explaination of Using LoRA and Prefix

Our hybrid method is mainly inspired by this paper (He et al., 2022a). Since the basic PEFT paradigms mainly include adapter, prefix, and LoRA, this paper analyzed the similarities and differences among them and concluded that fine-tuning methods parallel to the FFN and adding virtual tokens before the attention mechanism yield better results, which is similar to LoRA and prefix. Therefore, we chose the LoRA and prefix method.

Additionally, the adapter method normally has significantly higher parameter counts compared to LoRA and prefix, yet only achieved average results in experiments, whereas LoRA and prefix have similar less tunable parameter counts. The results between these three methods on six mathematical reasoning datasets using LLaMA-7B can be found in table 10.

Method	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
GPT-J-6B(1000)							
Prefix	32.50	25.49	56.96	53.64	61.76	8.33	39.78
LoRA	34.00	3.92	53.16	61.82	64.71	**9.09**	37.78
HLPT	**39.00**	3.92	**65.82**	**70.91**	63.73	**9.09**	42.08
H^2LPT	**39.00**	**27.45**	**65.82**	68.18	**68.63**	6.06	**45.86**
GPT-J-6B(500)							
Prefix	21.00	**29.41**	43.04	10.91	50.98	**6.44**	26.96
LoRA	28.00	1.96	**59.49**	24.55	**59.80**	3.79	29.60
HLPT	**35.00**	23.53	58.23	**32.73**	56.86	**6.44**	**35.46**
H^2LPT	29.00	3.92	49.37	27.27	53.92	3.03	27.75

Table 6: The experimental results of the model using the GPT-J model with dataset sizes of 1k and 0.5k.(**bold**: the best score; underline: the second best)

Method	ARC-c	ARC-e	Boolq	WinoG	PIQA	SIQA	OBQA	Avg.
Prefix	15.61	17.13	0.03	0.24	25.73	8.75	5.60	10.44
LoRA	23.89	28.54	**62.17**	49.49	**56.26**	39.71	31.80	41.69
HLPT	**38.99**	**55.60**	43.46	**50.91**	54.84	**50.72**	**43.40**	**48.27**
H^2LPT	36.01	55.18	60.86	49.33	48.42	45.85	41.60	48.18

Table 7: The extended experiments on commonsense reasoning datasets using GPT-J-6B (**bold**: the best score)

Method	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
LLaMA-7B							
(5:27)	51.50	9.80	70.89	78.18	78.43	18.56	51.23
(8:24)	59.00	17.65	68.35	80.00	81.37	26.14	55.42
(11:21)	58.50	13.73	74.68	84.55	83.33	19.70	55.75
(16:16)	60.50	15.69	78.48	89.09	83.33	25.38	58.75
(21:11)	62.50	25.49	74.68	89.09	82.35	20.08	59.03
(24:8)	62.50	25.49	78.48	90.91	77.45	23.86	59.78
(27:5)	59.00	17.65	75.95	91.82	77.45	25.00	57.81

Table 8: The specific results of figure 2 (a) on six different mathematical reasoning datasets.

Method	SVAMP	AQuA	AddSub	MultiArith	SingleEQ	GSM8K	Avg.
LLaMA-7B							
original	62.50	25.49	78.48	90.91	77.45	23.86	59.78
method1	58.50	23.53	75.95	85.45	84.31	20.08	57.97
method2	52.00	29.41	82.28	85.45	76.47	21.21	57.80
method3	59.50	27.45	72.15	88.18	78.43	21.21	57.82
method4	49.00	35.29	77.22	84.55	74.51	22.73	57.22
25%layers	52.50	15.69	73.42	80.82	83.33	23.11	54.81

Table 9: The specific results of figure 2 (c) and the extended experiments on six different mathematical reasoning datasets.

Method	Params	SVAMP	AQuA	AddSub	MultiA	SingleEQ	GSM8K	Avg.
LoRA	5.24M	58.50	23.53	75.95	92.73	88.24	24.24	60.53
Adapter	201.33M	53.50	23.53	74.68	86.36	75.49	20.08	55.61
prefix	7.86M	42.50	23.53	58.23	60.00	66.67	15.91	44.47

Table 10: Comparison of LoRA, Prefix, and Adapter methods using the LLaMA-7B model on mathematical reasoning datasets.