Not All LoRA Parameters Are Essential: Insights on Inference Necessity

Anonymous ACL submission

Abstract

Current research on LoRA primarily focuses on minimizing the number of fine-tuned parameters or optimizing its architecture. However, the necessity of all fine-tuned LoRA layers during inference remains underexplored. In this paper, we investigate the contribution of each LoRA layer to the model's ability to predict the ground truth and hypothesize that lower-layer LoRA modules play a more critical role in model reasoning and understanding. To address this, we propose a simple yet effective method to enhance the performance of large language models (LLMs) fine-tuned with LoRA. Specifically, we identify a "boundary layer" that distinguishes essential LoRA layers by analyzing a small set of validation samples. During inference, we drop all LoRA layers beyond this boundary. We evaluate our approach on three strong baselines across four widely-used text generation datasets. Our results demonstrate consistent and significant improvements, underscoring the effectiveness of selectively retaining critical LoRA layers during inference.

1 Introduction

Large language models (LLMs), such as ChatGPT, have demonstrated remarkable capabilities across diverse downstream tasks. However, their extensive parameterization presents substantial challenges for fine-tuning. In response, parameter-efficient fine-tuning (PEFT) methods (Houlsby et al., 2019; Li and Liang, 2021) have gained significant traction. Notably, Low-Rank Adaptation (LoRA) has emerged as a pivotal technique, particularly in the context of LLMs. LoRA operates by introducing trainable adapters for each layer of the LLM while keeping the remaining parameters frozen. This approach not only substantially reduces the computational resources required for fine-tuning but also achieves performance that is on par with or even superior to fully fine-tuned LLMs.

To further leverage LoRA for improving training efficiency and model performance, various studies have focused on optimizing its architecture or pruning important parameters for each layer. Zhang et al. (2023a) introduced AdaLoRA, which utilizes singular value decomposition (SVD) of $\triangle W$ to dynamically adjust the rank of LoRA for different layers. LoRA-Drop (Zhou et al., 2024) prunes LoRA parameters based on output evaluation. HydraLoRA (Tian et al., 2024) proposes an asymmetric structure that employs a shared A matrix and multiple B matrices to handle complex domain datasets. MoELoRA (Luo et al., 2024) selects suitable A and B matrices in LoRA for each layer to improve adaptation. However, these methods either focus on more efficient parameter fine-tuning or necessitate the fine-tuning of more complex LoRA structures. We argue that selectively using the fine-tuned LoRA of part of the layers without additional training will be more efficient, given the already small size of LoRA parameters.

In this work, we conduct a systematic investigation into the layer-wise impact of LoRA (Low-Rank Adaptation) in LLMs. Our empirical analysis reveals a distinct functional separation across model layers: the lower layers predominantly engage in content understanding and information extraction, while the upper layers specialize in answer summarization and refinement. Interestingly, our findings demonstrate that the top layers can effectively generate responses based on the representations captured by the bottom layers even without LoRA adaptation, leveraging the inherent knowledge encoded in the pre-trained LLMs. Building upon these observations, we propose a novel and computationally efficient approach to optimize LoRA-based fine-tuning. Specifically, we introduce two complementary strategies for identifying a critical "boundary layer" that separates information extraction from answer refinement in LoRA-enhanced models. The first strategy em-

ploys a manual approach by computing the average probability of ground truth outputs across LoRA-tuned layers using validation samples, with the boundary determined through probability curve analysis. The second strategy adopts an automated approach by evaluating model performance across different boundary layer configurations and selecting the optimal one. During inference, we strategically remove LoRA layers above the identified boundary layer, achieving both efficiency gains and performance maintenance.

We evaluate our proposed method on four widely-used generation datasets spanning multiple tasks, employing three state-of-the-art baselines: **Phi-2** (Li et al., 2023), **Llama2-7B-Chat** (Touvron et al., 2023), and **Llama-3.1-8B-Instruct** (Dubey et al., 2024). Our empirical results consistently demonstrate the effectiveness and generaliz ability of the proposed approach across different model architectures and task domains.

2 Related Work

2.1 Parameter-Efficient Fine-Tuning

With the rapid scaling of large language models (LLMs), traditional full-parameter fine-tuning becomes progressively impractical due to exponentially increasing computational costs. Consequently, parameter-efficient fine-tuning (PEFT) techniques have assumed greater significance (Houlsby et al., 2019). There are two main paradigms for PEFT based on their principles: prompt-based tuning and adapter-based methods.

Prompt-based techniques optimize the model through input-space interventions rather than architectural changes. Early implementations like Prompt Tuning (Lester et al., 2021) learn continuous task-specific embeddings prepended to input sequences, significantly reducing the computation of the parameters. Prefix-tuning (Li and Liang, 2021) optimizes virtual token embeddings across all transformer layers, demonstrating improved capability on generation tasks. Kwon et al. (2024) introduces adaptive proximal policy optimization to revise the stability and environment dependence. Although the Prompt-based techniques exhibit effective utilizations of few-shot and zero-shot data, they are still affected by their sensitivity to initialization and sequence length constraints.

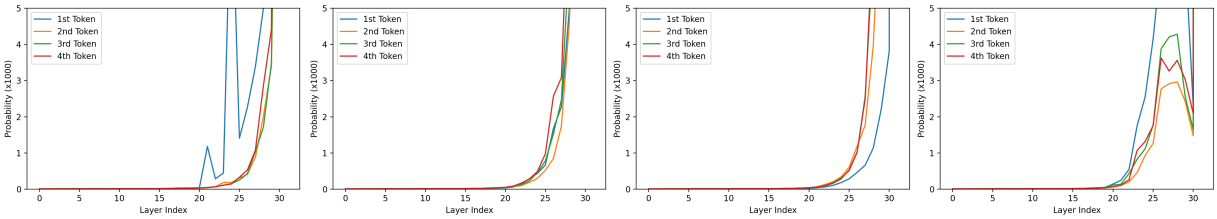
Adapter-based approaches introduce small trainable modules between transformer layers, achieving parameter efficiency through freezing the base model. Preliminary methods (Houlsby et al., 2019; Mahabadi et al., 2021; Bapna and Firat, 2019; Wang et al., 2021) introduce notable inference latency due to sequential computation bottlenecks. Low-Rank Adaptation (LoRA) further reduces the computational overhead through decomposing weight updates into low-rank matrices, achieving comparable performance to full fine-tuning with few parameters. Subsequent studies have extended this paradigm. AdaLoRA (Zhang et al., 2023b) dynamically allocates rank budgets across layers. LoRS (Hu et al., 2025) adopts weight recompute and computational graph rearrangement to reduce memory and computational consumption while improving performance. Moreover, Gao et al. (2024) observed that higher transformer layers require more LoRA experts to capture task-specific patterns, while lower layers exhibit significant redundancy. Hu et al. (2024) enhance parameter efficiency by sharing the LoRA A matrix across all layers, further proof the significant parameter redundancy of conventional LoRA architecture.

2.2 Knowledge Distillation

In contrast to the PEFT paradigm that focuses on optimizing model adaptation efficiency, an alternative trajectory explores knowledge distillation techniques for computational cost compression through inter-model knowledge transfer, which effectively reduces consumption, but generally faces the problem of a lack of task generalization abilities (Hahn and Choi, 2019; Sun et al., 2019; Tang et al., 2019). Tf-FD (Li, 2022) enhances generalization through layer-wised self-distillation and student features reusing. However, the limited ability of the student model to represent complex tasks results in insufficient adaptability. Liu et al. (2023) improves classification accuracy by linear transformation from N to one, but neglects cross-layer semantic discrepancies and multilevel knowledge integration. However, our approach takes a fundamentally different way to further unleash the vintrinsic knowledge of LLMs, leveraging the information captured by the LoRA of the bottom layers while allowing the model to reason in its original, unmodified manner at the top layers without LoRA.

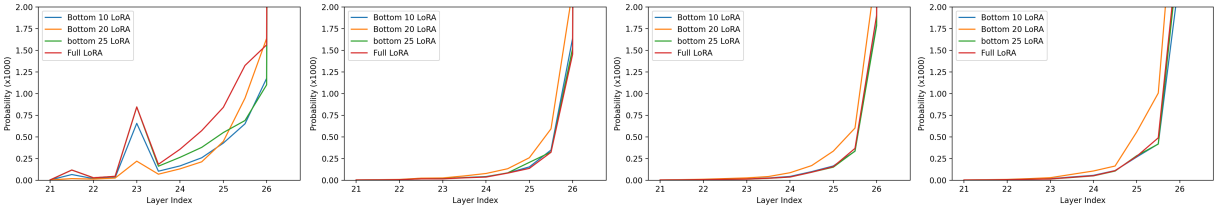
3 Preliminary Analysis of LoRA

In these preliminary analysis experiments, we use all three strong baselines fine-tuned with LoRA on four generation datasets in different tasks to explore



(a) HotpotQA (b) Samsum (c) GSM8K (d) WMT23 (EN->ZH)

Figure 1: The average maximum probability of the first four tokens for each layer of Llama3.1-8B-Instruct model fine-tuned with LoRA on the four datasets.



(a) 1st Token (b) 2nd Token (c) 3rd Token (d) 4th Token

Figure 2: The average maximum probability of the first four tokens for each layer of Llama3.1-8B-Instruct model fine-tuned with LoRA on the HotpotQA dataset while dropping specific LoRA layers during inference.

the impact of LoRA for different layers.

3.1 The Probability of Each Layer

We randomly sampled 100 instances from the test set of our four datasets. Then we utilize fine-tuned LLMs to output the probability distribution across different layers. By encoding the ground truth for each sample and averaging the corresponding probabilities, we assessed whether the models arrived at the correct answers. For enhanced clarity in our analysis, we visualized the probability distributions of the first several tokens at each layer. Figure 1 and 6 illustrate these distributions for the Llama3.1-8B-Instruct and Llama2-7B-Chat across the four datasets.

Observation Our analysis reveals a distinct pattern in the probability distribution across the layers of LLMs. Specifically, the bottom layers exhibit relatively low and stable probabilities for first four tokens. However, at a certain "boundary layer," we observe a sharp and significant increase in these probabilities. We posit that this transition reflects the model's progression from context comprehension and information extraction in the lower layers to answer formulation and refinement in the upper layers. This interpretation is supported by the visualization in Figure 6, which demonstrates that the initial layers maintain consistently low probabilities, followed by a marked upward trend in the bottom layers. This abrupt shift suggests that the model begins to synthesize the extracted information and generate task-appropriate responses beyond this "boundary layer".

3.2 Top LoRA Are Not Necessary

Building on the conclusions drawn in Section 3.1, we posit that for downstream tasks, it is crucial for LLMs to understand the context and effectively capture information. Imposing rigid formatting constraints may negatively impact model performance, given that LLMs are primarily pre-trained for text completion on natural language corpora. To investigate the layers of LoRA that are essential, we conduct experiments by selectively dropping specific LoRA of layers from an LLM. Specifically, we randomly select 100 samples from the test set and visualize the average maximum probability of the first four tokens at each layer when keeping the LoRA of the bottom 10, 20, and 25 layers. The resulting curves are depicted in Figure 2. We plot the average maximum probability of the same token across various scenarios in each sub-figure.

Observation Dropping the LoRA of the specific numbers of top layers does not significantly affect the output probabilities. As illustrated in Figure 2, keeping the LoRA of bottom layers does not significantly affect the model's output probabilities too much, particularly for the 2nd to 4th tokens (Figure 2b, 2c, 2d). This observation is reasonable, as the first token typically determines the

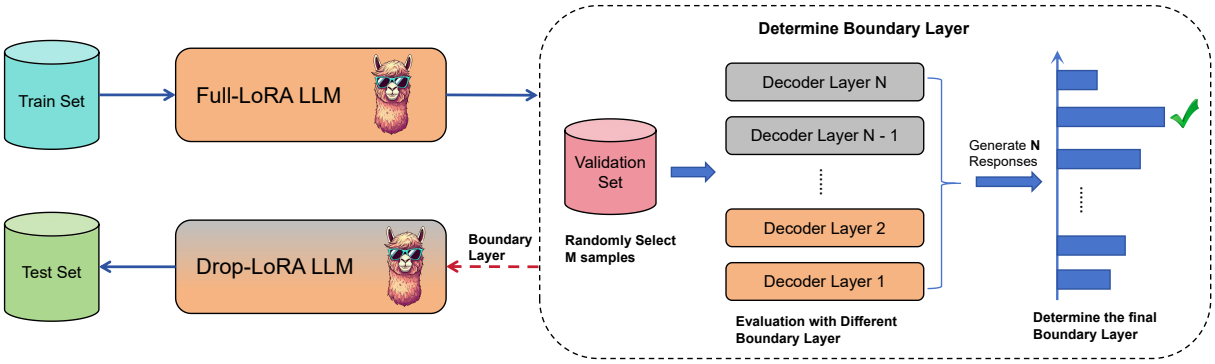


Figure 3: The overview of our proposed method.

response pattern for downstream tasks, making it more susceptible to fluctuations compared to subsequent tokens. This aligns with the findings of Zhan et al. (2024), which suggests that fine-tuning primarily impacts the first token. Furthermore, we observe that when keeping the LoRA of the bottom 20 layers (depicted by the orange curve), the model's maximum output probabilities outperform those in other configurations, including the baseline. Notably, even for the first token's probability, this "boundary layer" also results in higher maximum output probabilities in the last several layers compared to the baseline.

4 Method

Building upon our hypothesis and the empirical observations detailed in Section 3, we propose a simple yet effective strategy to improve model performance without requiring additional fine-tuning. Our approach centers on the removal of LoRA components from layers situated above an identified "boundary layer." A comprehensive overview of this methodology is illustrated in Figure 3.

Initially, we apply LoRA to all layers to conduct supervised fine-tuning (SFT) of a large language model (LLM) on a downstream task, adhering to standard procedures. Subsequently, we identify the "boundary layer" where LoRA should be removed. As analyzed in Section 3, a computationally efficient method to determine this layer involves calculating the average probability of the ground truth from the logits output of each layer and visualizing these probabilities. The point where the curve begins to rise is manually identified as the "boundary layer." After removing the LoRA from layers beyond this point, we can directly generate responses for the test set without further training.

Additionally, we propose a more precise and automated method to identify the "boundary layer," as illustrated in Figure 3. After fine-tuning the LLMs with full LoRA, we randomly select a set of M samples from the validation set to evaluate model performance after dropping LoRA at different "boundary layers." For instance, in the case of the **Llama3.1-8B-Instruct** model, which has 32 decoder layers, we obtain 32 evaluation results corresponding to different "boundary layers." We then select the best-performing result as our final "boundary layer" and evaluate the model performance on the test set after removing LoRA beyond this point. By using validation set samples to determine the boundary layer and subsequently evaluating on the test set, we mitigate the risk of overfitting the boundary layer to a specific dataset, thereby enhancing the credibility of our results.

5 Experiments

5.1 Setup

Dataset We use four widely used datasets in different generation tasks in our experiments: **HotpotQA** (Yang et al., 2018) for multi-document question answering, **GSM8K** (Cobbe et al., 2021) for mathematical reasoning, **Samsum** (Gliwa et al., 2019) for text summarization, and **WMT23** (Kocmi et al., 2023) for machine translation. The statistic about them is shown in Table 2.

Evaluation Metric We employ different evaluation metrics for these four tasks seperately. We employ Exact Match (EM) and F1 score to evaluate the performance of the HotpotQA dataset, whose purpose is to measure how completely the generated response contains the label. For **GSM8K**, we only use the EM score to evaluate whether the final calculated results is correct. For **Samsum** and **WMT23**, we use the traditional ROUGE-L and

	HotpotQA		GSM8K	Samsum		WMT23(En->Zh)
Model	EM	F1	EM	ROUGE	LLM-Score	BLEU
Phi-2 (Fine-tuned)	62.8	**70.0**	57.1	**33.7**	71.2	**6.4**
Phi-2 (Ours)	**65.6**	68.4	**57.8**	32.5	**72.5**	5.4
Llama2-7B-Chat (Fine-tuned)	66.2	**73.5**	36.1	42.8	78.0	27.0
Llama2-7B-Chat (Ours)	**69.1**	71.6	**38.6**	**43.5**	**80.3**	**27.4**
Llama3.1-8B (Fine-tuned)	73.1	**80.4**	73.1	**38.2**	79.5	33.0
Llama3.1-8B (Partial Fine-tuned)	73.0	79.9	73.9	37.7	79.8	33.3
Llama3.1-8B (Ours)	**74.1**	80.3	**74.3**	38.1	**80.6**	**35.8**

Table 1: The results (%) on the test set of the five datasets in our experiments. **Bold** numbers indicate the better result for each baseline.

Dataset	Train	Validation	Test
HotpotQA	50,000	7,405	7,405
GSM8K	7,473	1,319	1,319
Samsum	14,732	818	819
WMT23	50,000	500	2,074

Table 2: The statistics of the four datasets we used in our experimental setting.

BLEU[1] (Post, 2018) seperately at first. However, these two tasks can have similar meanings in different expressions. Thus, we also design a LLM-Score (Appendix A.2) to evaluate the generation output from various perspectives, such as fluency, accuracy, or readability, by asking the LLMs to act as a human.

Model We evaluate our method on three strong baselines: **Phi-2**[2] (Li et al., 2023), **Llama2-7B-Chat**[3] (Touvron et al., 2023), and **Llama-3.1-8B-Instruct**[4] (Dubey et al., 2024). More details of implementation will be shown in Appendix A.3.

5.2 Main Results

As shown in Table 1, our method of selectively dropping LoRA after specific layers outperforms the baseline in nearly all experiments. For the HotpotQA dataset, our method achieves a higher EM score. However, it consistently underperforms all three baselines in terms of F1 scores. This suggests that our approach is more effective in generating responses that contain the correct answers compared to the baselines. As discussed in our previous analysis, the LoRA components in the lower layers primarily help the LLM learn to format answers according to specific downstream tasks. When these LoRA components are dropped, the model continues to reason like the original model, often generating answers in natural language. This behavior leads to responses that include additional words alongside the correct answer, thereby negatively impacting the F1 scores.

For the GSM8K and WMT23 datasets, our method mostly outperforms the baselines, further verifying that removing LoRA from several top layers can enhance the reasoning capability of LLMs. It should be noted that the performance of our method on the Samsum dataset drops slightly on two baselines while achieving a better LLM-Score. This is because our approach deals with summarization tasks that are highly open-ended, and our method reduces the constraints imposed by the LoRA of top layers on the LLMs' inference capabilities, allowing the generated summary to be more comprehensive. However, as a trade-off, it also loses the ability of the top LoRA to mimic training data for formatting the answers, leading to a decrease in Rouge scores. Regarding the decline in performance on the WMT23 test set with Phi-2, our analysis revealed that the training data remains too challenging for Phi-2. We believe this is why removing LoRA in Phi-2 can not get the same improvement as the other two models.

5.3 Ablation Study

In this work, we present a robust and unified approach that resists decomposition into discrete components for analysis. However, our strategy of selectively dropping LoRA beyond the "boundary

[1] https://github.com/mjpost/sacrebleu
[2] https://huggingface.co/microsoft/phi-2
[3] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
[4] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

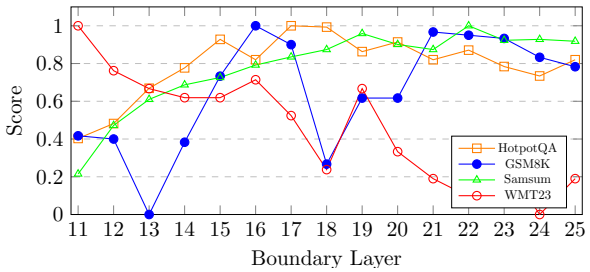


Figure 4: The performance of different "boundary layer" of Llama3.1-8B-Instruct model. The **Score** means the corresponding automatic evaluation metric of four datasets.

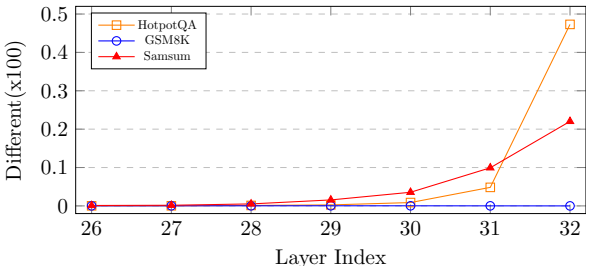


Figure 5: The probability difference of ground truth between our method and baseline on three datasets.

layer" during inference introduces a challenge: the need for a two-step process involving fine-tuning followed by LoRA removal. We conducted experiments across four datasets using Llama3.1-8B-Instruct, applying LoRA only to the bottom 20 layers of the LLMs during the fine-tuning phase. The evaluation results are shown in Table 1, which is the row of "Partial Fine-tuned". Interestingly, the performance of LLMs fine-tuned with LoRA in the bottom 20 layers closely matches that of the baseline models. This suggests that, regardless of whether LoRA is applied to all layers during fine-tuning, some LoRA components are particularly effective at capturing and understanding context, while others excel at synthesizing and refining answers to suit downstream tasks. These findings highlight the necessity of our two-step process.

6 Analysis

6.1 Different Boundary Layer

To provide a more intuitive rationale for our selection of the "boundary layer", we conduct additional experiments by removing the LoRA with different "boundary layers". Specifically, we assessed model performance after dropping the LoRA after the K=$[10, 25]$ layers, with the results illustrated in Figure 4. Consistently, removing the LoRA after the 15-20 layer resulted in better performance across all four datasets. This finding highlights the robustness of our approach and suggests that the "boundary layer" for one model is likely to remain a range across different downstream tasks.

6.2 Post-Hoc Analysis

Furthermore, we conduct some experiments to investigate how dropping LoRA affecting the LLMs performance and verify its effect. Specifically, we evaluated the probability distribution of the first token generated by our method and baseline models across three datasets: HotpotQA, Samsum, and GSM8K. Specifically, we extracted and compared the average probability of the ground truth token for all decoding layers. Typically, the average probability of the ground truth token for the first token is relatively low, making visualization challenging. To address this, we calculated the difference in average probabilities for each layer between our method and the baseline, and visualized the results in Figure 5. The visualization reveals that after dropping certain LoRA components, our method tends to assign higher probabilities to the ground truth token, especially near the output layers. It is important to note that the probability differences for GSM8K remain stable. This stability arises because, in GSM8K, the model must perform reasoning before arriving at the final answer, resulting in low probabilities for the first token in both our method and the baseline. This contrasts with HotpotQA, where the ground truth is relatively shorter, leading to the largest probability differences. A case study will be introduced in Appendix A.1 to concretely show the effect of our method.

6.3 Comparison of Labeling and Generation

Besides evaluating the performance of generation tasks, we also conduct analysis to verify whether our method can improve the performance on labeling tasks, where LLMs also get significant performance. The HotpotQA dataset can be partially viewed as a labeling task, featuring two types of questions: Bridge and Comparison. The answers to comparison questions are typically "Yes", "No", or an choice between "A" or "B", which is similar to the binary classification. We begin by assessing the performance improvements on these tasks. As depicted in Table 3, our approach yields EM scores for both question types, albeit with a reduction in F1 scores, mirroring trends observed in our main

Question Type	EM	F1
Bridge	78.1	**81.7**
Bridge (Ours)	**79.7**	80.4
Comparison	71.9	**79.8**
Comparison (Ours)	**72.7**	78.7

Table 3: The results of different types of question (%) HotpotQA dataset with Llama3.1-8B-Instruct fine-tuned with LoRA. **Bold** numbers indicate the best results.

Method	Accuracy (%)
E-Commerce (Fine-tuned)	68.6
E-Commerce (Ours)	**69.8**

Table 4: The accuracy (%) of our method and baseline on Llama3.1-8B-Instruct fine-tuned with E-Commerce dataset. **Bold** numbers indicate the best results.

	Samsum	WMT23(EN->ZH)
LLMs-based Evaluation		
Baseline	37.6	44.2
Ours	**62.4**	**55.8**
Real Human Evaluation		
Baseline	10.0	43.3
Ours	**90.0**	**56.7**

Table 5: The LLMs-based and real human evaluation on Llama3.1-8B-Instruct. This percentage(%) refers to the proportion of all test data where the output of this method is better than that of another method. **Bold** numbers indicate the best results.

results (Section 5.2). In particular, our method shows approximately double the improvement in EM scores for bridge compared to comparison.

Besides HotpotQA, we further evaluate our method using a purely labeling dataset. We fine-tune the LLaMA3.1-8B-Instruct with LoRA on a subset of the **E-Commerce** dataset (Zhang et al., 2018), selecting 50,000/500/2,000 data for the train/validation/test set, equally divided between positive and negative samples. This dataset focuses on dialogue response selection (Chen et al., 2024), requiring models to ascertain the suitability of a given response within a dialogue context. We utilize accuracy as the performance metric, preprocessing the data to align with LLM natural language formats with "Yes" or "No" labels for fine-tuning and evaluation. The results, presented in Table 4, indicate that our method surpasses the baseline by 1.2%, suggesting its efficacy for classification tasks. In addition, the output for this data should always be a single word. However, we observed that 13% of samples exceeded one word after dropping LoRA, with the additional content providing explanatory reasoning for the label. This observation supports our hypothesis that certain top LoRA are primarily involved in refining and formatting responses.

6.4 Generation Quality

Generally speaking, directly modifying some parameters of a fine-tuned LLM without continuing training may result in the generated content being incoherent or garbled. Thus, we further design two strategies to evaluate the generation quality of our method on **Samsum** and **WMT23**, since these two tasks are open-ended generation tasks and place greater emphasis on the quality of the generated results. The first one is similar to LLM-Score, which we ask LLMs to act as humans to choose the better one between our method and baselines. The other is real human evaluation. We randomly sample 30 examples seperately in test set of **Samsum** and **WMT23** datasets, and recruited three computer science students proficient in both English and Chinese to choose the better approach between our method and the baseline method based on four aspects: fluency, accuracy, readability, and completeness. Then, we calculated the average results from these three students. As shown in Table 5, the percentage(%) represents the proportion of all test data where the evaluators considered the results of this method to be better.

The output from our method gets a higher proportion consistently, whether evaluated by LLMs or real humans. It indicates that our method can improve the model performance without affecting the generation quality through only dropping some LoRA during inference. The evaluation instruction for LLMs will be shown in Table 9. In comparison to text summarization, our method did not achieve as significant results in translation tasks. We think it is because the prediction results in translation tasks are more strongly constrained by the source sentence, which means that the gains from dropping LoRA of specific layers are relatively smaller. This also indicates that our method performs better in tasks with fewer constraints on the ground truth format, such as open-ended generation tasks.

	HotpotQA		GSM8K	Samsum		WMT23(En->Zh)
Method	EM	F1	EM	ROUGE	LLM-Score	BLEU
Baseline ($r = 16$)	73.9	**80.1**	74.4	**38.6**	80.4	32.9
Ours ($r = 16$)	**74.6**	**80.1**	**76.4**	38.0	**80.7**	**36.8**
Baseline ($r = 32$)	72.9	79.9	75.8	38.5	80.8	32.6
Ours ($r = 32$)	**73.6**	**80.0**	**76.0**	**38.6**	**81.2**	**36.0**

Table 6: The results of different LoRA rank on Llama3.1-8B-Instruct. **Bold** numbers indicate the best results.

6.5 Impact of LoRA Rank

One of the pivotal factors influencing the performance of LoRA tuning is the choice of LoRA rank. As different rank values can lead to varying performances even with identical models and datasets, we conducted a comprehensive experimental evaluation on the Llama3.1-8B-Instruct model. Specifically, we examined rank values of $\{16, 32\}$, with the detailed results presented in Table 6. Our empirical findings demonstrate that our method consistently outperforms nearly all baseline approaches, thereby providing strong evidence for its generalization capabilities. Furthermore, our analysis revealed that while overfitting phenomena were observed on the GSM8K and WMT23 datasets, as indicated by performance degradation, dropping top LoRA can effectively mitigate this issue and enhance model performance. It means that our method possesses inherent mechanisms that can alleviate overfitting to a significant extent, making it more robust across diverse datasets and tasks.

6.6 Robustness of OOD Data

Building on the insights from Section 6.5, we further investigate our method's robustness through out-of-domain (OOD) evaluation. To this end, we construct two test sets by randomly sampling 2,000 en→zh parallel sentences from **WikiMatrix** (Schwenk et al., 2021) (general domain) and **ParaMed** (Liu and Huang, 2021) (biomedical domain). These datasets are evaluated on Llama3.1-8B-Instruct models fine-tuned exclusively on WMT23 (news domain), using the same boundary layer configuration. As shown in Table 7, our method demonstrates consistent superiority over baselines despite potential domain mismatch, suggesting its strong generalization capabilities even when the optimal boundary layer may vary across domains.

	WIKI	PARAM
Baseline	19.9	22.6
Ours	**21.0**	**24.9**

Table 7: The BLEU score of different domain translation data on Llama3.1-8B-Instruct fine-tuned with News data. **Bold** numbers indicate the best results.

6.7 Error Analysis

In this section, we examine cases where our proposed method underperforms across all experimental setups. Our results in Table 1 indicate a consistent decline in F1 scores compared to the baseline. Thus, we conducted a detailed error analysis on the HotpotQA dataset, focusing on samples where our method yielded lower F1 scores. Our analysis reveals two key observations: (1) In 56% of these cases, the output generated by our method fully contains the baseline's output, and (2) when both methods produce correct answers, this overlap increases to 96%. This suggests that our approach tends to generate longer and more comprehensive responses compared to the baseline. We hypothesize that this behavior stems from the influence of top LoRA in LLMs. These LoRA appear to bias the model toward producing more structured or formatted outputs. When these layers are removed during inference, the output probability distribution becomes smoother, potentially leading to more verbose and inclusive responses.

7 Conclusion

In this work, we investigate the impact of fine-tuning LoRA across various layers in LLMs, revealing that LoRA beyond a specific "boundary layer" becomes redundant during inference. We propose a simple yet effective method to enhance LoRA-fine-tuned LLM performance. Future directions include leveraging these findings to develop advanced techniques for optimizing LoRA efficacy.

8 Limitation

Our approach requires sampling a subset from the validation set to identify the boundary layer, introducing additional computational cost, although the boundary layer can be directly set to $k = \{15, 20\}$ as a practical alternative based on our analysis. Our experimental results highlight that the method imposes specific requirements on the capability of LLMs: the model must possess sufficient capacity to effectively learn the downstream task. Otherwise, the LoRA of all layers may be utilized during inference to compensate for insufficient task-specific knowledge. For example, a model like Phi-2, which achieves only a 6.4 BLEU score on translation tasks due to inadequate learning, would significantly undermine the efficacy of our approach.

References

AI Anthropic. 2024. Claude 3.5 sonnet model card addendum. *Claude-3.5 Model Card*, 3(6).

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Guanhua Chen, Runzhe Zhan, Derek F. Wong, and Lidia S. Chao. 2024. Dynamic curriculum learning for conversation response selection. *Knowledge-Based Systems*, 293:111687.

Yew Ken Chia, Liying Cheng, Hou Pong Chan, Chaoqun Liu, Maojia Song, Sharifah Mahani Aljunied, Soujanya Poria, and Lidong Bing. 2024. M-longdoc: A benchmark for multimodal super-long document understanding and a retrieval-aware tuning framework. *arXiv preprint arXiv:2411.06176*.

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*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Baochen Sun, Ruibo Liu, Daiyi Peng, Yawen Zhang, Xiaoyuan Guo, Jie Yang, and VS Subrahmanian. 2024. Higher layers need more lora experts. *Preprint*, arXiv:2402.08562.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Sangchul Hahn and Heeyoul Choi. 2019. Self-knowledge distillation in natural language processing. *arXiv preprint arXiv:1908.01851*.

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.

Junyan Hu, Xue Xiao, Mengqi Zhang, Yao Chen, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. 2024. Aslora: Adaptive sharing low-rank adaptation across layers. *Preprint*, arXiv:2412.10135.

Yuxuan Hu, Jing Zhang, Xiaodong Chen, Zhe Zhao, Cuiping Li, and Hong Chen. 2025. Lors: Efficient low-rank adaptation for sparse large language model. *Preprint*, arXiv:2501.08582.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Philipp Koehn, Benjamin Marie, Christof Monz, Makoto Morishita, Kenton Murray, Makoto Nagata, Toshiaki Nakazawa, Martin Popel, Maja Popović, and Mariya Shmatova. 2023. Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In *Proceedings of the Eighth Conference on Machine Translation*, pages 1–42, Singapore. Association for Computational Linguistics.

Minchan Kwon, Gaeun Kim, Jongsuk Kim, Haeil Lee, and Junmo Kim. 2024. Stableprompt: Automatic prompt tuning using reinforcement learning for large language models. *Preprint*, arXiv:2410.07652.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *Preprint*, arXiv:2104.08691.

Lujun Li. 2022. Self-regulated feature learning via teacher-free feature distillation. In *European Conference on Computer Vision*, pages 347–363. Springer.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: **phi-1.5** technical report. *arXiv preprint arXiv:2309.05463*.

Boxiang Liu and Liang Huang. 2021. Paramed: A parallel corpus for english–chinese translation in the biomedical domain. *BMC Medical Informatics and Decision Making*, 21(1):258.

Xiaolong Liu, Lujun Li, Chao Li, and Anbang Yao. 2023. Norm: Knowledge distillation via n-to-one representation matching. *Preprint*, arXiv:2305.13803.

Tongxu Luo, Jiahe Lei, Fangyu Lei, Weihao Liu, Shizhu He, Jun Zhao, and Kang Liu. 2024. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *arXiv preprint arXiv:2402.12851*.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Preprint*, arXiv:2106.04647.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Holger Schwenk, Vishrav Chaudhary, Shuo Sun, Hongyu Gong, and Francisco Guzmán. 2021. WikiMatrix: Mining 135M parallel sentences in 1620 language pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Raphael Tang, Yao Lu, and Jimmy Lin. 2019. Natural language generation for effective knowledge distillation. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 202–208.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. 2024. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *arXiv preprint arXiv:2404.19245*.

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

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Runzhe Zhan, Xinyi Yang, Derek Wong, Lidia Chao, and Yue Zhang. 2024. Prefix text as a yarn: Eliciting non-English alignment in foundation language model. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12131–12145, Bangkok, Thailand. Association for Computational Linguistics.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023a. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*. Openreview.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023b. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *Preprint*, arXiv:2303.10512.

Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, and Hai Zhao. 2018. Modeling multi-turn conversation with deep utterance aggregation. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 3740–3752.

Hongyun Zhou, Xiangyu Lu, Wang Xu, Conghui Zhu, and Tiejun Zhao. 2024. Lora-drop: Efficient lora parameter pruning based on output evaluation. *arXiv preprint arXiv:2402.07721*.

A Appendix

A.1 Case Study

We analyze several random samples concretely to explore the effect of our proposed method. The most notable results are observed with the **Samsum** dataset, because the summarization task is very flexible, where multiple valid expressions can effectively capture the essence of a paragraph. The examples provided in Table 10, after applying our method to drop specific LoRA, the LLM generates responses that are more comprehensive and

accurate. This observation aligns with our initial hypothesis: the LoRA of the top several layers primarily functions to format responses according to downstream task, which may, to some extent, constrain the model's reasoning capabilities. By removing these specific LoRA, the LLM is able to leverage the knowledge captured with the bottom LoRA to engage in more divergent reasoning.

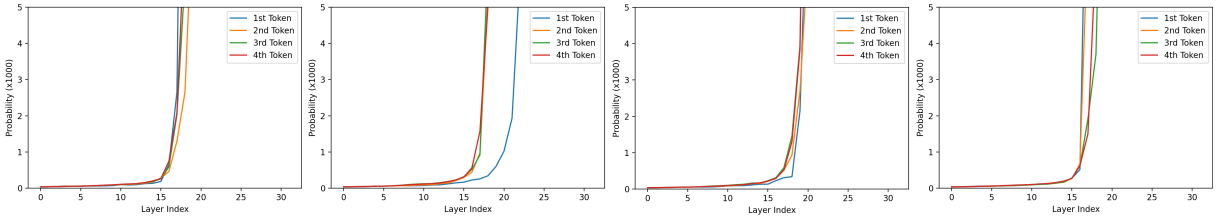
A.2 LLM-Score

For this metric, three powerful LLMs are employed as the evaluators following Chia et al. (2024). We use the API of GPT-4o (Hurst et al., 2024), claude-3-5-sonnet (Anthropic, 2024), and Gemini-1.5-pro (Team et al., 2024) to generate the responses. The instruction to ask the LLMs to output the scores for the predicted results is shown in Table 8. We use the percentage(%) format of the average scores of these three LLMs as the final scores in our experiments.

A.3 Implementation

We fine-tuned all the models in our experiments with **LLaMA-Factory**[5]. We fine-tuned these three models 3 epochs for all datasets. And we set the learning rate to $1e-4$ and LoRA rank to 8. The maximum length of each sample and the batch size are set to 2048 and 16. We sample $M = 500$ instances from the validation set to determine the "boundary layer", and we set "boundary layer" to 15 directly if a dataset does not have a validation set. All the experiments have been completed on one 80G H800 GPU or vGPU-32GB.

[5]https://github.com/hiyouga/LLaMA-Factory



(a) HotpotQA (b) Natural Question (c) GSM8K (d) WMT23 (EN->ZH)

Figure 6: The average maximum probability of each layer of Llama2-7B-Chat model fine-tuned with LoRA on the four datasets.

Instruction of LLM-Score
Here is the given dialogue history: **{Dialog History}** Here is the ground truth: **{Label Response}** Here is the predicted output: **{The Response Generated by LLMs}** - The aim of this prediction is to generate a summary of the provided historical dialogue. - Please rate the summary based on its level of abstraction and fluency in summarizing the historical dialogue. The scores should be integers ranging from 0 to 10. - Please give the score number directly without any explanation or introduction.

Table 8: The instruction of calculating the LLM-Score in our experiments for **Samsum** dataset. For translation tasks, we only need to adjust the task description.

Instruction of Evaluating the Generation Quality
Here is the given dialogue history: **{Dialog History}** Here are two generated responses: A. **{Baseline}** B. **{Our Method}** - The aim of these responses is to generate a summary of the provided historical dialogue. - Please choose the better response like a human based on four aspects: fluency, accuracy, readability, and completeness. - Please output only "A" or "B" directly without any explanation or introduction.

Table 9: The instruction of evaluating the generation quality for **Samsum** dataset. "**A**" is the output from baselines and "**B**" is the output from our method. For translation tasks, we also need to adjust the task description.

An Example in Samsum	
Eric: MACHINE!	**Rob**: That's so gr8!
Eric: I know! And shows how Americans see Russian	**Rob**: And it's really funny!
Eric: I know! I especially like the train part!	**Rob**: Hahaha! No one talks to the machine like that!
Eric: Is this his only stand-up?	**Rob**: Idk. I'll check.
Eric: Sure.	**Rob**: Turns out no! There are some of his stand-ups on youtube.
Eric: Gr8! I'll watch them now!	**Rob**: Me too!
Baseline: Eric and Rob enjoy watching MACHINE stand-up. **Ours**: Eric and Rob are watching a Russian comedian's stand-up. They will watch more of his videos on YouTube.	

Table 10: An example of the test set of Samsum dataset.