Understanding Layer Significance in LLM Alignment

Guangyuan Shi[1], Zexin Lu[1], Xiaoyu Dong[1], Wenlong Zhang[1], Xuanyu Zhang[2], Yujie Feng[1], Xiao-Ming Wu[1]✉
[1]The Hong Kong Polytechnic University, Hong Kong S.A.R., China
[2]Xiaohongshu Inc., China
{guang-yuan.shi, zexin.lu, xiaoyu.dong, wenlong.zhang, yujie.feng}
@connect.polyu.hk, xyz@mail.bnu.edu.cn, xiao-ming.wu@polyu.edu.hk

Abstract

Aligning large language models (LLMs) through supervised fine-tuning is essential for tailoring them to specific applications. Recent studies suggest that alignment primarily adjusts a model's presentation style rather than its foundational knowledge, indicating that only certain components of the model are significantly impacted. To uncover how alignment affects model behavior at a granular level, we propose identifying which layers within LLMs are most critical to the alignment process. Our approach, named ILA, involves learning a binary mask for the parameter changes in each layer during alignment, as an indicator of layer significance. Experimental results reveal that, despite substantial differences in alignment datasets, the important layers of a model identified by ILA exhibit nearly 90% overlap, highlighting fundamental patterns in LLM alignment. The results also indicate that freezing non-essential layers improves overall model performance, while selectively tuning the most critical layers significantly enhances fine-tuning efficiency with minimal performance loss. Finally, we discuss how these findings extend from LLM alignment to reasoning. The source code is available at https://github.com/moukamisama/ILA.

1 Introduction

Aligning large language models (LLMs) with specific requirements is essential for enhancing their utility across diverse applications (Luo et al., 2023a; Yu et al., 2023; Luo et al., 2023b; Li et al., 2023; Liu et al., 2024a; 2022; Feng et al., 2023). Fine-tuning LLMs during the alignment process can significantly improve the models' capabilities to meet targeted needs (Bubeck et al., 2023). Typically, alignment involves fine-tuning the model on diverse datasets, which may include both human-curated (Rajani et al., 2023) and LLM-generated (Taori et al., 2023) data, using approaches like instruction tuning (Wei et al., 2021) and preference learning (Bai et al., 2022; Rafailov et al., 2024). Given the significant cost associated with full parameter fine-tuning, parameter-efficient fine-tuning (PEFT) (Hu et al., 2021; Chen et al., 2022; Pan et al., 2024) methods have emerged as a popular alternative, offering a balance between performance and resource efficiency.

Understanding what LLMs actually learn during the alignment process is crucial. Zhou et al. (2023) posits that the majority of knowledge and capabilities are developed during the pre-training phase, with alignment primarily serving to refine the model's conversational style and formatting. Using a well-selected set of 1,000 training examples for supervised fine-tuning (SFT), they successfully produced a high-quality aligned model. Similarly, Lin et al. (2023) investigated the token distribution of LLMs before and after alignment and found that most changes were related to "stylistic tokens", such as discourse markers and transition words, while the knowledge-intensive content largely remained untouched, coming from the base pre-trained model. These findings imply that the alignment process mainly adjusts the model's presentation style rather than modifying its foundational knowledge.

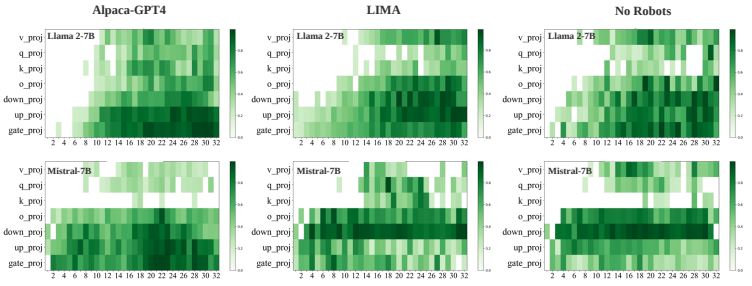

Figure 1: Layer importance rankings by our ILA algorithm for LLAMA 2-7B and Mistral-7B-v0.1 across Alpaca-GPT4, LIMA, and No Robots datasets. **Top 75% layers by score (s_i) are considered important**. X-axis: transformer block index; y-axis: linear layer names. The figure highlights two findings: (1) **High overlap (90%)** in important layers across datasets (Table2) suggests shared alignment needs, regardless of substantial differences in dataset content; (2) Important layers **differ by architecture**, reflecting model-specific dynamics.

Table 1: Impact of fine-tuning different components of LLAMA 2-7B on alignment performance using the LIMA dataset. Evaluated on MMLU (5-shot) and GPT-4o scores for Vicuna and MT-Bench prompts. Tuned components include attention projections (W_q, W_k, W_v, W_o) and feed-forward layers (W_{up}, W_{down}, W_{gate}).

	ATT (W_q, W_k, W_v, W_o)	**ATT2** (W_q, W_k, W_v)	**FFN** (W_{up}, W_{down}, W_{gate})	**ALL** (LoRA)
MMLU ↑	42.03	42.65	43.06	**43.18**
Vicuna ↑	5.21	5.13	5.40	**5.43**
MT-Bench ↑	3.31	3.35	3.41	**3.45**

To gain a deeper understanding of LLM alignment, we analyze this process at the level of model parameters. We conducted a pilot study to investigate the impact of different model components on alignment performance, by fine-tuning only specific layers and evaluating the resulting performance, as presented in Table 1. The results clearly indicate that fine-tuning different components of the LLM leads to considerable performance differences. For instance, fine-tuning the feed-forward network (FFN) layers achieves performance similar to fine-tuning all linear layers (i.e., with LoRA), whereas focusing solely on the attention layers causes a notable drop in performance. This observation shows the complexity of layer-specific contributions to LLM alignment, highlighting the need for detailed analysis.

To address this, we propose ***identifying the layers that are most critical to alignment performance during the SFT process***. We develop a novel approach, ILA, for identifying the important layers for LLM alignment. Specifically, we learn a binary mask for the parameter changes in each layer during the fine-tuning process, which serves as an indicator of layer significance. A binary mask value of zero indicates that the corresponding layer has negligible influence during the process, while a value of one denotes that the layer is crucial. We use gradient descent to learn the binary mask effectively and offer a theoretical analysis of the optimization process. The main findings and significance of this work include:

- **Consistent layer importance ranking across different alignment datasets.** We observe similar rankings of important layers during alignment for the same pre-trained model (see Fig. 1), even though the alignment datasets vary significantly in both content and size. This suggests that the alignment process endows the model with similar capabilities, corroborating previous research findings and offers new insights into LLM alignment.
- **Enhancing performance by freezing unimportant layers.** We show that freezing about 25% of unimportant layers can improve performance and that *a single search* for layer importance ranking is sufficient for different alignment tasks using the same architecture.
- **Improving alignment efficiency through selective fine-tuning.** Our findings show that fine-tuning only 10-30% key layers achieves performance comparable to fine-tuning all

linear layers. Additionally, integrating this approach with QLoRA allows tuning only 30-75% of key layers to maintain or enhance performance while cutting resource costs.

- **Broader implications beyond LLM alignment.** Although our primary focus is on LLM alignment, the approaches and insights from this study have broader applicability. Our preliminary experiments on LLM reasoning reveal findings similar to those in alignment, showcasing the significant potential of our methods to enhance the reasoning capabilities of LLMs, particularly in achieving test-time scaling (OpenAI, 2024; Welleck et al., 2024; Snell et al., 2024; Muennighoff et al., 2025).

2 Related Works

LLM Alignment. Pretrained language models encode general-purpose representations, enabling transfer across diverse tasks (Qiu et al., 2024; Jiang et al., 2024; Nijkamp et al., 2022). Alignment methods like instruction tuning (Zhang et al., 2023c; Sun et al., 2023; Muennighoff et al., 2023) and preference learning (Hejna et al., 2023; Guan et al., 2022; Rafailov et al., 2024; Song et al., 2024; Li et al., 2024a) adapt these models to specific objectives. Recent studies have explored alignment mechanisms. LIMA (Zhou et al., 2023) showed that fine-tuning on small datasets (e.g., 1,000 examples) shapes behavior without adding new knowledge, a finding echoed by others (Chen et al., 2023; Lee et al., 2023; Gudibande et al., 2023). Duan et al. (2023) connected instruction tuning to in-context learning via hidden state analysis, while URIAL (Lin et al., 2023) revealed that alignment mainly modifies stylistic tokens, preserving knowledge-centric ones. These insights suggest alignment imparts narrow, targeted adjustments. Our work builds on this by identifying the specific layers most critical for alignment, offering a more fine-grained understanding of how adaptation occurs.

Parameter Efficient Fine-Tuning (PEFT). Fine-tuning large language models with billions or trillions of parameters is computationally expensive (Brown et al., 2020; Fedus et al., 2022). Parameter-efficient fine-tuning (PEFT) methods address this by updating specific components (Zaken et al., 2021; Zhao et al., 2020; Ansell et al., 2021; Guo et al., 2020) or using soft prompts (Lester et al., 2021; Li & Liang, 2021; Asai et al., 2022). Techniques such as BitFit (Zaken et al., 2021), Adapters (Houlsby et al., 2019), LoRA (Hu et al., 2021), and their variants (Zhang et al., 2023b; Meng et al., 2024) reduce cost while maintaining transferability. Recent work (Li et al., 2024b; Hui et al., 2024; Pan et al., 2024; Xu & Zhang, 2024; Panda et al., 2024) shows that selectively fine-tuning certain regions yields strong results, though random masking often lacks consistency. However, most PEFT approaches overlook parameter importance and lack prioritization. Our method addresses this by ranking layer importance, enabling targeted fine-tuning to improve performance with minimal cost.

Layer Analysis in Model Compression. Model compression techniques use structured pruning (Xia et al., 2022; Liu et al., 2024b; van der Ouderaa et al., 2023) and layer analysis to improve efficiency. Methods like Sheared LLaMA (Xia et al., 2023) and LLM-Streamline (Chen et al., 2024) show that pruning layers, heads, and dimensions can significantly reduce model size with minimal performance loss. Layer importance studies (Zhang et al., 2024; Gromov et al., 2024) further support the removal of less critical components for scalability. However, these efforts focus on reducing model size rather than optimizing parameter updates for task-specific alignment. In contrast, our work targets alignment fine-tuning by prioritizing parameter updates through skill localization (Panigrahi et al., 2023; Voita et al., 2023), improving both alignment efficiency and robustness.

3 Quantifying Layer Significance in LLM Alignment

To understand layer significance in LLM alignment, we propose ILA, a method to identify important layers by learning a binary mask that indicates each layer's significance.

Consider a pre-trained LLM model with parameters θ_0 composed of N layers, i.e., $\theta_0 = \{\theta_0^i\}_{i=1}^N$. The model is fine-tuned on an alignment dataset $\mathcal{D} = \{z_i\}_{i=1}^n$ with a loss function $\mathcal{L}(\theta)$. After t training iterations, the model parameters are updated to $\theta_t = \theta_0 + \Delta\theta_t$, where $\Delta\theta_t$ represents the change in parameters till iteration t. Define a binary mask $\gamma_t = \{\gamma_t^i | \gamma_t^i \in$

Algorithm 1: Identify the Important Layers for Alignment (ILA)

Input: Pre-trained model parameters θ_0, learning rate α, the initial importance score vector $\boldsymbol{s}_0 = \{s_0^i\}_{i=1}^N$, the number of insignificant layers K, the low-rank matrices A_0, B_0 for the LoRA algorithm. (FFT is a special case of LoRA with full rank)

for iteration $i = 1, 2, \ldots$ **do**
  Update $A_t = A_{t-1} - \alpha \nabla_{A_{t-1}} \mathcal{L}(\theta_t)$, $B_t = B_{t-1} - \alpha \nabla_{B_{t-1}} \mathcal{L}(\theta_t)$ (LoRA);
  Or Update $\theta_t = \theta_{t-1} - \alpha \nabla_{\theta_{t-1}} \mathcal{L}(\theta_{t-1})$ (FFT);
  if *Training has become stable* **then**
    Solve the optimization problem in Eq. (7) by gradient descent to find $\boldsymbol{s}_t = \{s_t^i\}_{i=1}^N$;
    Stop training;
  end
end

$\{0,1\}\}_{i=1}^N$ that encodes layer-wise importance information. We apply $\boldsymbol{\gamma}_t$ to $\Delta\theta_t$ and define

$$\theta_t^{\text{mask}} = \theta_0 + \boldsymbol{\gamma}_t \odot \Delta\theta_t, \tag{1}$$

where $\odot$ is component-wise multiplication. The binary mask is applied to retain the changes in crucial layers while eliminating the rest. Below we provide a formal definition of the conditions under which training attains stability after an adequate number of iterations.

Definition 1 (ϵ-stable). *$\forall \epsilon > 0$, the model is said to be ϵ-stable at iteration T if, for any $t > T$, the loss function satisfies the condition*

$$|\mathbb{E}_z[\mathcal{L}(z; \theta_{t+1})] - \mathbb{E}_z[\mathcal{L}(z; \theta_t)]| < \epsilon, \tag{2}$$

where $\mathbb{E}_z[\cdot]$ denotes the expectation with respect to the alignment dataset $\mathcal{D}$.

Once training stabilizes, we can identify the layers that are crucial for the alignment task.

Definition 2 (Layer Importance). *The binary mask $\boldsymbol{\gamma}_t$ is defined as the solution to the following optimization problem:*

$$\boldsymbol{\gamma}_t = \arg\min_{\boldsymbol{\gamma}_t} \mathbb{E}_z[\mathcal{L}(z; \theta_t^{\text{mask}})], \; s.t. \; \|\boldsymbol{\gamma}_t\| < H, \tag{3}$$

where H is a hyper-parameter that serves as a constraint to limit the number of important layers.

Efficiently Identifying the Importance Layers (Alg. 1). Due to the high cost of fine-tuning large models, to address the optimization problem in Eq. (3), we employ the LoRA (Hu et al., 2021) algorithm, which utilizes low-rank decomposition matrices to represent the change in model parameters till iteration t ($\Delta\theta_t$). Specifically, LoRA utilizes two trainable low-rank matrices, $\boldsymbol{B}_t^i \in \mathbb{R}^{d_i \times r_i}$ and $\boldsymbol{A}_t^i \in \mathbb{R}^{r_i \times k_i}$, to estimate the change of the i^{th} layer:

$$\Delta\theta_t^i = \beta \cdot \boldsymbol{B}_t^i \boldsymbol{A}_t^i, \tag{4}$$

where β is the scalar hyperparameter of LoRA. With the binary mask $\boldsymbol{\gamma}_t$, the i^{th} layer is updated by

$$\theta_t^i = \theta_0^i + \beta \cdot \gamma_t^i \cdot \boldsymbol{B}_t^i \boldsymbol{A}_t^i. \tag{5}$$

To ease the optimization of $\boldsymbol{\gamma}_t$, we re-parametrize each of its each components γ_t^i as the output of a Sigmoid function, i.e., $\gamma_t^i = \sigma(s_t^i)$. Then, the update of the i^{th} layer becomes

$$\theta_t^i = \theta_0^i + \beta \cdot \sigma(s_t^i) \cdot \boldsymbol{B}_t^i \boldsymbol{A}_t^i. \tag{6}$$

Let $\boldsymbol{s}_t = \{s_t^i\}_{i=1}^N$, $\theta_t^{\text{M}} = \{\theta_t^i\}_{i=1}^N$. The optimization problem in Eq. (3) becomes

$$\boldsymbol{s}_t = \arg\min_{\boldsymbol{s}_t} \mathbb{E}_z[\mathcal{L}(z; \theta_t^{\text{M}})]. \tag{7}$$

We use gradient descent to optimize $\boldsymbol{s}_t$, yielding s_t^i as the importance score of the i^{th} layer. A larger value of s_t^i indicates γ_t^i is closer to one, signifying higher importance of the i^{th} layer.

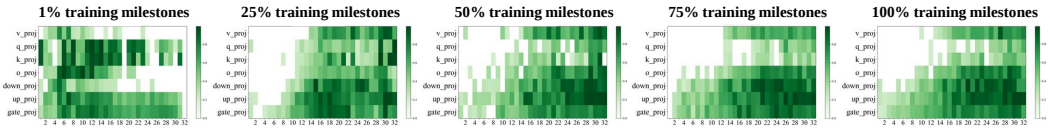

Figure 2: Layer importance rankings of LLAMA 2-7B during fine-tuning on LIMA at 1%, 25%, 50%, 75%, and 100% milestones. X-axis: transformer block index; y-axis: linear layer names. Jaccard similarities are provided in Table 4.

Assumption 3.1 (Lipschitz-continuous)**.** *The loss function $\mathcal{L}(\theta) : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable and L-smooth with constant $L_1 > 0$ such that*

$$\|\mathcal{L}(\theta) - \mathcal{L}(\theta')\|_2 \leq L_1\|\theta - \theta'\|. \tag{8}$$

In addition, $\mathcal{L}(\theta)$ has an L-Lipschitz continuous gradient with constant $L_2 > 0$ such that

$$\|\nabla\mathcal{L}(\theta) - \nabla\mathcal{L}(\theta')\|_2 \leq L_2\|\theta - \theta'\|. \tag{9}$$

Assumption 3.2. *For any $t > T$, θ_t is ϵ-stable. We assume there is a constant R such that*

$$\|\theta_t - \theta_{t+1}\|_2 \leq R\epsilon, \tag{10}$$

and there is a constant Q such that $\|\theta_t\|_2 \leq Q$ for any $t > T$.

Theorem 3.1. *Let $\epsilon > 0$ be sufficiently small such that θ_T is ϵ-stable. Suppose Assumptions 3.1 and 3.2 hold. For any $t > T$, we assume that $\forall i, \gamma_t^i \in [0, 1]$. Let γ_t' denote the result of γ_t after one step of gradient descent, i.e., $\gamma_t' = \gamma_t - \beta\nabla_{\gamma_t}\mathcal{L}(\theta_t^{\text{mask}})$. Then we have*

$$\|\gamma_t' - \gamma_{t+1}'\|_2 \leq \beta(QL_2 + L_1)R\epsilon. \tag{11}$$

This theorem demonstrates that when θ_T is ϵ-stable, solving the optimization problem in Eq. (3) for any $t > T$ yields similar results. This is because, after one step of gradient descent, the difference between γ_t and γ_{t+1} is smaller than a sufficiently small number. The proof is provided in Appendix A, and empirical results supporting this are shown in Fig. 2.

Leveraging Layer Importance Rankings. The identified rankings of layer importance can be leveraged to enhanc both the performance and efficiency of LLM alignment. To maximize performance, prioritize fine-tuning the significant layers while freezing those deemed less important. For efficiency, focus on the layers most critical to model success. Detailed experiments and analyses are presented in Sec. 4.

4 Experiments and Findings

4.1 Experimental Setup

Datasets. We use three alignment datasets—Alpaca-GPT4 (Peng et al., 2023), LIMA (Zhou et al., 2023), and No Robots (Rajani et al., 2023). See Appendix B for details.

Models and Baselines. We use LLAMA 2-7B/13B (Touvron et al., 2023), Llama 3.1-8B (Dubey et al., 2024), and Mistral-7B-v0.1 (Jiang et al., 2023). Baselines include LoRA (Hu et al., 2021), AdaLoRA (Zhang et al., 2023a), and full fine-tuning. See Appendix B for details.

Evaluation. We evaluate (1) language understanding using MMLU (Hendrycks et al., 2021) and Hellaswag (Zellers et al., 2019), and (2) conversational ability using MT-Bench (Zheng et al., 2023) and Vicuna (Chiang et al., 2023), with scoring conducted by **GPT-4o**. Further details are provided in Appendix B.

4.2 Layer Importance Rankings in LLM Alignment

In this subsection, we applied ILA to rank important layers during alignment across three datasets—No Robots, LIMA, and Alpaca-GPT4 (Fig.1). We also analyzed layer importance

Table 2: Jaccard similarities of the top 75% highest-scoring layers identified as important during fine-tuning of LLAMA 2-7B and Mistral-7B on various datasets.

Datasets	LLAMA 2-7B			Mistral-7B		
	LIMA	No Robots	Alpaca-GPT4	LIMA	No Robots	Alpaca-GPT4
LIMA	-	-	-	-	-	-
No Robots	0.91	-	-	0.90	-	-
Alpaca-GPT4	0.90	0.90	-	0.89	0.93	-

Table 3: Jaccard similarities of the top 75% important layers in LLAMA 2-7B fine-tuned on the LIMA dataset using different random seeds.

Random Seed	seed1	seed2	seed3
seed1	-	-	-
seed2	0.92	-	-
seed3	0.91	0.91	-

Table 4: Jaccard similarities of the top 75% important layers identified at different stages of LLAMA 2-7B fine-tuning on the LIMA dataset.

Training Milestones	1%	25%	50%	75%	100%
1%	-	-	-	-	-
25%	0.69	-	-	-	-
50%	0.70	0.91	-	-	-
75%	0.69	0.90	0.92	-	-
100%	0.69	0.91	0.92	0.93	-

rankings at different training milestones (Fig.2). To quantify similarity between sets of important layers, we used the Jaccard similarity coefficient, defining the top 75% highest-scoring layers as the important set S. The similarity between two sets, S_1 and S_2, is given by: $J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$. where $J = 1$ indicates identical sets, and $J = 0$ indicates no overlap.

Consistency in Layer Importance Rankings Across Different Datasets. Our findings show strong consistency in layer importance rankings: (1) highly similar important layers are identified across different alignment datasets, (Fig.1, Table2); (2) the rankings remain stable across different random seeds for γ (Table 3); and (3) similar layers can be identified even at the beginning stages of training, such as completion of 25% (Fig.2, Table4).

These results confirm the robustness of ILA, which consistently identifies stable and overlapping layers across datasets. This aligns with recent findings that alignment largely involves stylistic token shifts (Lin et al., 2023). In essence, alignment seeks similar capabilities, as *evidenced by our observation that important layers remain stable across different datasets*. This underscores the relevance of our algorithm to the fundamental objectives of alignment.

Feed-Forward Network (FFN) layers are more important than attention layers. Our layer ranking results (Fig. 1) consistently demonstrate that Feed-Forward Network (FFN) layers are more important than attention layers. This observation aligns with previous studies (Geva et al., 2020; Meng et al., 2022), which identify FFNs as key memory modules responsible for encoding factual knowledge, while attention layers primarily facilitate token interactions. This distinction explains why FFN layers consistently achieve higher importance rankings in our results. Moreover, our findings suggest several practical implications: (1) Model compression should prioritize retaining upper FFN layers. (2) Knowledge editing may be more effective when targeting middle-to-upper FFN layers.

4.3 Enhancing Alignment Performance through Freezing Unimportant Layers

To leverage layer importance rankings, we excluded less important layers that could negatively impact fine-tuning, removing approximately 25% of unimportant layers. The main results on **No Robots** are in Table 5, with additional results for LLAMA 2-13B (see Table 18) and main results on Alpaca-GPT4 (see Table 20), and LIMA (see Table 19) datasets in Appendix C.5. Key observations include:

(1) **Freezing unimportant layers can enhance performance**. ILA consistently outperformed LoRA and full fine-tuning on most metrics, with freezing 25% of unimportant layers yielding better results than tuning all layers. (2) **A single search for layer importance ranking suffices for a given architecture.** Rankings were stable across alignment tasks, allowing us to compute it on the No Robots dataset and apply it to others.

Table 5: Comparison of LLAMA 2-7B, Mistral-7B-v0.1, and Llama 3.1-8B fine-tuned on the No Robots dataset, evaluated on **MMLU** (5-shot), **Hellaswag** (0-shot), and **GPT-4o scores** for **Vicuna** and **MT-Bench** prompts. Vicuna and MT-Bench results are averaged over **three runs**. Grey cells indicate improvements over the base model; best scores are in bold.

Models	Methods	Language Understanding		Conversational Ability	
		MMLU ↑	Hellaswag ↑	Vicuna ↑	MT-Bench ↑
LLAMA 2-7B	AdaLoRA	45.23	57.30	5.81	4.01
	Full Fine-tune	45.72	57.69	6.12	4.18
	Full Fine-tune w/ ILA	**45.98**	57.87	**6.35**	**4.37**
	LoRA	44.58	59.46	5.78	4.02
	LoRA w/ ILA	45.78	**59.65**	5.90	4.33
Mistral-7B-v0.1	AdaLoRA	62.13	61.68	6.21	4.69
	Full Fine-tune	61.05	**64.26**	6.32	4.55
	Full Fine-tune w/ ILA	61.75	64.21	**6.51**	4.78
	LoRA	61.95	62.90	6.25	4.68
	LoRA w/ ILA	**62.14**	62.98	6.42	**4.87**
Llama 3.1-8B	AdaLoRA	64.85	62.85	6.51	5.08
	Full Fine-tune	64.44	63.65	6.50	5.11
	Full Fine-tune w/ ILA	65.00	**63.69**	**6.61**	**5.23**
	LoRA	64.95	60.77	6.33	4.58
	LoRA w/ ILA	**65.43**	60.95	6.45	4.69

These results show that ILA improves fine-tuning efficiency by focusing on significant layers. Compared to AdaLoRA, even though we explored a narrow range of the hyperparameter t_r (target average rank of incremental matrices), our method performed better, suggesting that adjusting LoRA's matrix rank alone doesn't guarantee superior results, as also noted in (Dettmers et al., 2023).

Additionally, as discussed in Sec. 4.2, the stability of the layer importance ranking across datasets means a single search is often sufficient. In our experiments, we computed the layer importance ranking using full training iterations on the No Robots dataset, and then directly applied this ranking to other datasets. Though dataset-specific rankings can further improve results (Table 16 in Sec. 6), the strong cross-dataset performance with one ranking demonstrates our approach's robustness and generalizability.

4.4 Enhancing Alignment Efficiency by Fine-tuning Only the Most Critical Layers

To investigate this issue, we fine-tune the top 10%, 20%, and 30% of the important layers of Mistral-7B-v0.1, as identified by ILA, on the No Robots dataset, and compare the results with those of the LoRA algorithm. The results demonstrate clear benefits in focusing on a subset of important layers:

(1) Fine-tuning a small subset of the most important layers achieves competitive performance and enhances efficiency. Fine-tuning the top 10%, 20%, or 30% of layers results in only a slight performance drop compared to full fine-tuning. Fine-tuning 30% of layers nearly matches full fine-tuning (Table 6), demonstrating that focusing on the most important layers ensures efficient fine-tuning with minimal performance loss.

(2) Our method can be applied to enhance QLoRA, further reducing costs. When combined with QLoRA, our method fine-tunes only 30% or 75% of the most important layers while maintaining or improving performance (Table 7), highlighting the efficiency of our approach in achieving comparable or better results with fewer layers.

These findings highlight the effectiveness of our layer selection strategy, optimizing resource use with minimal performance trade-offs. Our integration with QLoRA shows that fine-tuning a targeted subset of important layers improves both performance and memory efficiency during fine-tuning.

Table 6: Fine-tuning results of Mistral-7B-v0.1 on the No Robots dataset, evaluated on MMLU (5-shot), Hellaswag (0-shot), and GPT-4o scores for Vicuna and MT-Bench prompts (averaged over three runs). Percentages in parentheses denote the fraction of fine-tuned linear layers. Best results are in bold.

Models	Methods	Language Understanding		Conversational Ability	
		MMLU ↑	Hellaswag ↑	Vicuna ↑	MT-Bench ↑
Mistral-7B-v0.1	LoRA	**61.95**	**62.90**	6.25	4.68
	LoRA w/ ILA (10%)	62.09	61.94	5.99	4.39
	LoRA w/ ILA (20%)	61.83	62.16	6.12	4.53
	LoRA w/ ILA (30%)	61.89	62.79	**6.27**	**4.75**

Table 7: Comparison of QLoRA fine-tuning on LLAMA 2-7B vs. selectively fine-tuning important layers identified by ILA. Evaluated on MMLU (5-shot), Hellaswag (0-shot), and GPT-4o scores for Vicuna and MT-Bench prompts (averaged over three runs). Grey cells indicate improvements over the base model by ILA.

Datasets	Methods	Language Understanding		Conversational Ability	
		MMLU ↑	Hellaswag ↑	Vicuna ↑	MT-Bench ↑
LIMA	QLoRA	43.06	55.47	5.31	2.98
	QLoRA w/ ILA (75%)	43.48	55.95	5.56	3.19
	QLoRA w/ ILA (30%)	44.01	55.82	5.17	3.01

For a clearer understanding of GPU memory savings, we measured memory consumption for QLoRA, LoRA, Full Fine-Tuning, and versions fine-tuning only the key layers identified by ILA. Results are presented in Table 15 in Appendix C.1.

4.5 Ablation Study

Observation 1: Our layer importance ranking algorithm is effective. We evaluated our algorithm by comparing it to a baseline that fine-tunes all layers and three alternatives: (1) **RL 1** and **RL 2**, which randomly freeze top-K layers; (2) **FL**, freezing the first K layers; and (3) **LL**, freezing the last K layers. As shown in Table 8, these naive strategies underperform. In contrast, our method effectively identifies and freezes the least critical layers, yielding notable gains in both efficiency and performance.

Observation 2: The important scores calculated using LoRA are similar to those obtained through full fine-tuning. To assess whether LoRA-based approximations differ from full fine-tuning (FFT), we compared parameter updates from LoRA (i.e., Eq. (4)) and FFT (i.e., $\Delta\theta_t = \theta_t - \theta_0$). For both methods, we derived layer importance scores and selected the top 75% of layers, then calculated the Jaccard similarity between the layers. As shown in Table 9, LoRA achieves nearly 83% overlap with the important layers identified by FFT, reducing computational overhead while effectively ranking layer importance. The results show that LoRA provides a strong approximation of $\Delta\theta_t$ compared to $\theta_t - \theta_0$.

Furthermore, we identified layers marked as important (Top 75%) by FFT but not by LoRA on LLaMA 2-7B using the No Robots dataset. These layers had low average ranks according to FFT (124 out of 168; best rank: 57), suggesting they are not among the most critical layers.

Observation 3: Cross-dataset evaluation of layer importance enhances performance. Different datasets highlight subtle differences in important layers (Table 2). By intersecting the top-K least important layers from LIMA, No Robots, and Alpaca-GPT4 and freezing them during fine-tuning (Table 16, Appendix C.2), we found that cross-dataset evaluation yields better results than dataset-specific fine-tuning. This suggests that assessing layer importance across datasets leads to more robust fine-tuning.

Observation 4: Cross-model transfer of layer importance rankings is feasible but less effective than cross-dataset transfer. Models sharing architecture but trained on different

Table 8: Performance comparison of ILA, random, and position-based layer selection for fine-tuning LLAMA 2-7B on the No Robots dataset. **RL1**/**RL2** freeze K randomly selected layers (different seeds); **FL** and **LL** freeze the first and last K layers, respectively. Blue highlights indicate lower performance than ILA.

Methods	Language Understanding		Conversational Ability	
	MMLU ↑	Hellaswag ↑	Vicuna ↑	MT-Bench ↑
LoRA	44.58	59.46	5.78	3.98
LoRA w/ RL 1	44.23	59.71	5.72	3.96
LoRA w/ RL 2	43.98	59.11	5.62	3.89
LoRA w/ FL	44.02	59.32	5.58	3.71
LoRA w/ LL	44.61	59.21	5.65	3.99
LoRA w/ ILA	45.78	59.65	5.90	4.15

Table 9: Jaccard Similarity between important layers selected using Full Fine-Tuning and LoRA for LLAMA 2-7B. Top 75% highest-scoring layers are determined as important layers.

Datasets	LIMA (FFT)	No Robots (FFT)	Alpaca-GPT4 (FFT)
LIMA (LoRA)	0.84	0.76	0.83
No Robots (LoRA)	0.78	0.80	0.81
Alpaca-GPT4 (LoRA)	0.82	0.83	0.86

Table 10: Jaccard similarities of the top 75% important layers across different models.

	LLAMA 2-7B (LIMA)	LLAMA 2-7B (No Robots)	LLAMA 2-7B (Alpaca-GPT4)
Mistral-7B-v0.1 (LIMA)	0.67	-	-
Mistral-7B-v0.1 (NoRobots)	0.70	0.71	-
Mistral-7B-v0.1 (Alpaca-GPT4)	0.71	0.66	0.75

Table 11: Experimental results for Mistral-7B-v0.1 on the No Robots dataset, using layer importance rankings derived from LLAMA 2-7B.

Methods	MMLU	Hellaswag	Vicuna	MT-Bench
LoRA	61.95	62.90	6.25	4.68
LoRA w/ ILA (75%) (cross-model transfer)	62.10	**63.21**	6.29	4.72
LoRA w/ ILA (75%)	**62.14**	62.80	**6.42**	**4.87**
LoRA w/ ILA (30%) (cross-model transfer)	61.77	**63.16**	6.11	4.60
LoRA w/ ILA (30%)	**61.89**	62.79	**6.27**	**4.75**

datasets show strong agreement in important layers (Jaccard similarity of 0.90 for the top 75%, Table 2). This drops to 0.70 across architectures (Table 10), indicating reduced transferability. Nonetheless, significant overlap suggests cross-architecture transfer remains viable. Fine-tuning Mistral-7B-v0.1 using rankings from LLAMA 2-7B on No Robots (Table 11) confirms that cross-model transfer can still perform well. Further analysis can be found in Appendix C.3.

Observation 5: ILA is robust to the initialization of layer importance scores. As demonstrated in Table 17 (Appendix C.4), our algorithm ILA is resilient to varying initial importance scores ($s_0 = 4.0, 2.0, 1.0$), with minimal impact on final rankings. The stable Jaccard similarities for the top 75% of layers during LLAMA 2-7B fine-tuning on LIMA confirm reliable convergence regardless of initialization.

Observation 6: The computation cost of ILA is low. ILA runs in two stages: Stage 1 trains the model with LoRA until ϵ-stability, and Stage 2 tunes importance weights (γ_t) with the backbone and LoRA frozen. For both LLAMA 2-7B and Mistral-7B-v0.1 (225 linear layers), Stage 1 takes 6671 ms per iteration, and Stage 2 takes 5343 ms. Stage 2 finishes in 11 minutes (128 batches). Most cost lies in Stage 1, but Table 4 shows only 25–50% of training milestones are needed for strong performance.

Table 12: Jaccard similarities of the top 75% important layers in LLAMA 2-7B.

	No Robots	LIMA	Alpaca-GPT4
OASST1	0.86	0.90	0.83

Table 13: Comparison of LLAMA 2-7B fine-tuned on the OASST1 dataset.

	MMLU	Hellaswag	Vicuna	MT-Bench
LoRA	44.82	58.98	6.62	4.46
LoRA w/ ILA (∼75%)	**45.24**	**59.22**	**6.75**	**4.51**

Observation 7: ILA performs well on datasets with conflicting stylistic objectives. We evaluated ILA on the OASST1 dataset (Köpf et al., 2023), which features diverse linguistic styles from contributors with varied backgrounds. Using a high-quality subset (as in QLoRA (Dettmers et al., 2023)), we fine-tuned LLAMA 22-7B and compared ILA's layer rankings with those from LIMA, No Robots, and Alpaca-GPT4. Table 12 shows high Jaccard Similarities (0.83–0.90), indicating consistent layer identification despite stylistic variation. These scores are slightly lower than the similarities among LIMA, No Robots, and Alpaca-GPT4 themselves (nearly 91%; see Table 2), which is expected due to OASST1's broader stylistic diversity. Fine-tuning only the top nearly 75% ILA-selected layers also yields consistent gains (see Table 13), confirming ILA's robustness to stylistic conflict.

5 Conclusion and Discussion: Beyond LLM Alignment

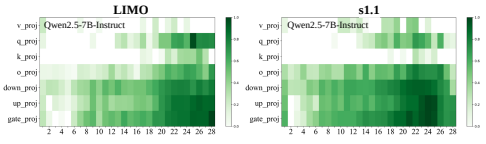

Figure 3: Layer-wise importance rankings for Qwen2.5-7B-Instruct fine-tuned using the LIMO and s1.1 datasets, respectively.

Table 14: Performance of Qwen2.5-7B-Instruct on mathematical reasoning benchmarks after fine-tuning with the LIMO dataset.

Methods	**MATH500**	**AIME**
FFT	77.00	13.33
FFT w/ ILA	**79.00**	**16.67**

Our findings indicate that the alignment process of LLM imparts similar capabilities despite data variations. This complements prior research by revealing layer-specific roles and improving efficiency through strategic tuning and freezing of layers. Nonetheless, the approaches and insights derived from this study extend beyond LLM alignment.

LLM Reasoning and Test-time Scaling. Advanced models like o1 (OpenAI, 2024), Deepseek R1 (Guo et al., 2025), and Kimi 1.5 (Team et al., 2025) have exhibited strong reasoning capabilities. Similar to alignment, reasoning seeks to *further activate the knowledge* acquired during pre-training. While alignment ensures that outputs align with human values, reasoning *drives the model toward deeper inference for enhanced accuracy*. Rather than scaling model size or training data, recent studies such as LIMO (Ye et al., 2025) and s1 (Muennighoff et al., 2025) investigate *test-time scaling*—boosting performance by increasing the number of input tokens used for reasoning. Their findings show that even limited high-quality training data with chain-of-thought (CoT) examples can effectively enhance LLMs' reasoning capabilities.

To investigate whether our methods can provide insights into LLM reasoning comparable to those obtained in the context of LLM alignment, we conducted a set of preliminary experiments. Specifically, we fine-tuned the Qwen2.5-7B-Instruct model (Yang et al., 2024) on two distinct reasoning datasets, S1 (Muennighoff et al., 2025) and LIMO (Ye et al., 2025). Our analysis shows that the layers deemed most important by ILA on S1 and LIMO share a Jaccard similarity of nearly ***86%*** (Fig.3), indicating that the model develops comparable reasoning capabilities despite differences in dataset characteristics. Furthermore, we find that freezing the least important 25% of layers consistently improves downstream performance (Table14), thereby **underscoring the effectiveness of selective fine-tuning as a strategy for reasoning-centric tasks**. Additional details regarding dataset construction, evaluation protocols, and benchmark selection are provided in Appendix D.

References

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer. *arXiv preprint arXiv:2110.07560*, 2021.

Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. Attempt: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. *arXiv preprint arXiv:2205.11961*, 2022.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

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

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. Revisiting parameter-efficient tuning: Are we really there yet? *arXiv preprint arXiv:2202.07962*, 2022.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*, 2023.

Xiaodong Chen, Yuxuan Hu, and Jing Zhang. Compressing large language models by streamlining the unimportant layer. *arXiv preprint arXiv:2403.19135*, 2024.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Ves Stoyanov, and Alexis Conneau. Self-training improves pre-training for natural language understanding. *arXiv preprint arXiv:2010.02194*, 2020.

Hanyu Duan, Yixuan Tang, Yi Yang, Ahmed Abbasi, and Kar Yan Tam. Exploring the relationship between in-context learning and instruction tuning. *arXiv preprint arXiv:2311.10367*, 2023.

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

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. Towards llm-driven dialogue state tracking. *arXiv preprint arXiv:2310.14970*, 2023.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts. The unreasonable ineffectiveness of the deeper layers. *arXiv preprint arXiv:2403.17887*, 2024.

Lin Guan, Karthik Valmeekam, and Subbarao Kambhampati. Relative behavioral attributes: Filling the gap between symbolic goal specification and reward learning from human preferences. *arXiv preprint arXiv:2210.15906*, 2022.

Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. The false promise of imitating proprietary llms. *arXiv preprint arXiv:2305.15717*, 2023.

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

Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020.

Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W Bradley Knox, and Dorsa Sadigh. Contrastive prefence learning: Learning from human feedback without rl. *arXiv preprint arXiv:2310.13639*, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

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

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

Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. Hft: Half fine-tuning for large language models. *arXiv preprint arXiv:2404.18466*, 2024.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Guanying Jiang, Lingyong Yan, Haibo Shi, and Dawei Yin. The real, the better: Aligning large language models with online human behaviors. *arXiv preprint arXiv:2405.00578*, 2024.

Andreas Köpf, Yannic Kilcher, Dimitri Von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36:47669–47681, 2023.

Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*, 2023.

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

Aaron J Li, Satyapriya Krishna, and Himabindu Lakkaraju. More rlhf, more trust? on the impact of human preference alignment on language model trustworthiness. *arXiv preprint arXiv:2404.18870*, 2024a.

Haoling Li, Xin Zhang, Xiao Liu, Yeyun Gong, Yifan Wang, Yujiu Yang, Qi Chen, and Peng Cheng. Gradient-mask tuning elevates the upper limits of llm performance. *arXiv preprint arXiv:2406.15330*, 2024b.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023.

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

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. The unlocking spell on base llms: Rethinking alignment via in-context learning. *arXiv preprint arXiv:2312.01552*, 2023.

Qijiong Liu, Jieming Zhu, Quanyu Dai, and Xiao-Ming Wu. Boosting deep ctr prediction with a plug-and-play pre-trainer for news recommendation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 2823–2833, 2022.

Qijiong Liu, Jieming Zhu, Yanting Yang, Quanyu Dai, Zhaocheng Du, Xiao-Ming Wu, Zhou Zhao, Rui Zhang, and Zhenhua Dong. Multimodal pretraining, adaptation, and generation for recommendation: A survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 6566–6576, 2024a.

Yijin Liu, Fandong Meng, and Jie Zhou. Accelerating inference in large language models with a unified layer skipping strategy. *arXiv preprint arXiv:2404.06954*, 2024b.

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

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

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *arXiv preprint arXiv:2404.02948*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Niklas Muennighoff, Qian Liu, Armel Zebaze, Qinkai Zheng, Binyuan Hui, Terry Yue Zhuo, Swayam Singh, Xiangru Tang, Leandro Von Werra, and Shayne Longpre. Octopack: Instruction tuning code large language models. *arXiv preprint arXiv:2308.07124*, 2023.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*, 2022.

OpenAI. Learning to reason with LLMs, September 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. Lisa: Layerwise importance sampling for memory-efficient large language model fine-tuning. *arXiv preprint arXiv:2403.17919*, 2024.

Ashwinee Panda, Berivan Isik, Xiangyu Qi, Sanmi Koyejo, Tsachy Weissman, and Prateek Mittal. Lottery ticket adaptation: Mitigating destructive interference in llms. *arXiv preprint arXiv:2406.16797*, 2024.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. Task-specific skill localization in fine-tuned language models. In *International Conference on Machine Learning*, pp. 27011–27033. PMLR, 2023.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4, 2023.

Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M Ponti, and Shay B Cohen. Spectral editing of activations for large language model alignment. *arXiv preprint arXiv:2405.09719*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

Nazneen Rajani, Lewis Tunstall, Edward Beeching, Nathan Lambert, Alexander M. Rush, and Thomas Wolf. No robots. https://huggingface.co/datasets/HuggingFaceH4/no_robots, 2023.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18990–18998, 2024.

Xianghui Sun, Yunjie Ji, Baochang Ma, and Xiangang Li. A comparative study between full-parameter and lora-based fine-tuning on chinese instruction data for instruction following large language model. *arXiv preprint arXiv:2304.08109*, 2023.

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

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

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

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

Tycho FA van der Ouderaa, Markus Nagel, Mart Van Baalen, Yuki M Asano, and Tijmen Blankevoort. The llm surgeon. *arXiv preprint arXiv:2312.17244*, 2023.

Elena Voita, Javier Ferrando, and Christoforos Nalmpantis. Neurons in large language models: Dead, n-gram, positional. *arXiv preprint arXiv:2309.04827*, 2023.

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

Sean Welleck, Amanda Bertsch, Matthew Finlayson, Hailey Schoelkopf, Alex Xie, Graham Neubig, Ilia Kulikov, and Zaid Harchaoui. From decoding to meta-generation: Inference-time algorithms for large language models. *arXiv preprint arXiv:2406.16838*, 2024.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. Structured pruning learns compact and accurate models. *arXiv preprint arXiv:2204.00408*, 2022.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

Jing Xu and Jingzhao Zhang. Random masking finds winning tickets for parameter efficient fine-tuning. *arXiv preprint arXiv:2405.02596*, 2024.

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

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

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

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

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*, 2023c.

Yang Zhang, Yanfei Dong, and Kenji Kawaguchi. Investigating layer importance in large language models. *arXiv preprint arXiv:2409.14381*, 2024.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. Masking as an efficient alternative to finetuning for pretrained language models. *arXiv preprint arXiv:2004.12406*, 2020.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. Lima: Less is more for alignment, 2023.

A Proof of Theorem 3.1

The proof of Theorem 3.1 is presented as follows:

Theorem A.1. *For a sufficiently small ϵ, θ_T is ϵ-stable, thus Assumption 3.1 and Assumption 3.2 are satisfied. For any $t > T$, we assume that $\forall i, \gamma_t^i \in [0, 1]$. Let γ_t' denote the result of γ_t after one step of gradient descent, i.e., $\gamma_t' = \gamma_t - \beta \nabla_{\gamma_t} \mathcal{L}(\theta_t^{\text{mask}})$. Then we have*

$$\|\gamma_t' - \gamma_{t+1}'\|_2 \leq \beta(QL_2 + L_1)R\epsilon. \tag{12}$$

Proof. Let $\hat{\gamma}$ be the initial values of γ_t and γ_{t+1}. Then we have

$$\gamma_t' = \hat{\gamma} - \beta \nabla_{\gamma_t} \mathcal{L}(\theta_t^{\text{mask}}), \tag{13}$$

$$\gamma_{t+1}' = \hat{\gamma} - \beta \nabla_{\gamma_{t+1}} \mathcal{L}(\theta_{t+1}^{\text{mask}}). \tag{14}$$

The difference of γ_t' and γ_{t+1}' is

$$\begin{aligned}
\|\gamma_t' - \gamma_{t+1}'\|_2 &= \|(\hat{\gamma} - \beta \nabla_{\gamma_t} \mathcal{L}(\theta_t^{\text{mask}})) \\
&\quad - (\hat{\gamma} - \beta \nabla_{\gamma_{t+1}} \mathcal{L}(\theta_{t+1}^{\text{mask}}))\|_2 \\
&= \beta \|\nabla_{\gamma_t} \mathcal{L}(\theta_t^{\text{mask}}) - \nabla_{\gamma_{t+1}} \mathcal{L}(\theta_{t+1}^{\text{mask}})\|_2 \\
&= \beta \|\theta_t \odot \nabla_{\theta_t^{\text{mask}}} \mathcal{L}(\theta_t^{\text{mask}}) \\
&\quad - \theta_{t+1} \odot \nabla_{\theta_{t+1}^{\text{mask}}} \mathcal{L}(\theta_{t+1}^{\text{mask}})\|_2 \\
&\leq \beta \|\theta_t \odot (\nabla_{\theta_t^{\text{mask}}} \mathcal{L}(\theta_t^{\text{mask}}) - \nabla_{\theta_{t+1}^{\text{mask}}} \mathcal{L}(\theta_{t+1}^{\text{mask}}))\|_2 \\
&\quad + \beta \|(\theta_t - \theta_{t+1}) \odot \nabla_{\theta_{t+1}^{\text{mask}}} \mathcal{L}(\theta_{t+1}^{\text{mask}})\|_2.
\end{aligned} \tag{15}$$

Because $\mathcal{L}(\theta)$ has an L-Lipschitz continuous gradient with constant $L_2 > 0$, and $\|\theta_t\| \leq Q$,

$$\begin{aligned}
&\|\theta_t \odot \nabla_{\theta_t^{\text{mask}}} \mathcal{L}(\theta_t^{\text{mask}}) - \theta_{t+1} \odot \nabla_{\theta_{t+1}^{\text{mask}}} \mathcal{L}(\theta_{t+1}^{\text{mask}})\|_2 \\
&\leq QL_2 \|\theta_t^{\text{mask}} - \theta_{t+1}^{\text{mask}}\|_2 \\
&= QL_2 \|\Delta\theta_{t+1} - \Delta\theta_t\|_2 \\
&= QL_2 \|\theta_{t+1} - \theta_t\|_2.
\end{aligned} \tag{16}$$

Because $\mathcal{L}(\theta)$ is L-smooth with constant L_1,

$$\|(\theta_t - \theta_{t+1}) \odot \nabla_{\theta_{t+1}^{\text{mask}}} \mathcal{L}(\theta_{t+1}^{\text{mask}})\|_2 \leq L_1 \|\theta_t - \theta_{t+1}\|. \tag{17}$$

Therefore,

$$\|\gamma_t' - \gamma_{t+1}'\|_2 \leq \beta(QL_2 + L_1)\|\theta_t - \theta_{t+1}\|_2. \tag{18}$$

According to the Assumption 3.2, we have $\|\theta_t - \theta_{t+1}\|_2 \leq R\epsilon$, hence,

$$\|\gamma_t' - \gamma_{t+1}'\|_2 \leq \beta(QL_2 + L_1)R\epsilon. \tag{19}$$

□

B Experimental Setup

Datasets. (1) Alpaca-GPT4 contains 52K instruction-following data generated by GPT-4, utilizing prompts from Alpaca (Taori et al., 2023). (2) LIMA contains only 1K carefully curated prompts and responses. (3) No Robots contains 10K instructions and demonstrations created by skilled human annotators.

Models and Baselines. We use four different models as the base for our experiments: LLAMA 2-7B (Touvron et al., 2023), LLAMA 2-13B, LLAMA 3.1-8B (Dubey et al., 2024), and Mistral-7B-v0.1 (Jiang et al., 2023). Our baselines are as follows: (1) **LoRA**(Hu et al., 2021): Trainable rank decomposition matrices are added in parallel to existing weight matrices, including query/key/value projection (W_q, W_k, W_v), output projection (W_o) in self-attention, feed-forward networks (W_{up}, W_{down}, W_{gate}), and the output layer (W_{head}). (2) **AdaLoRA**(Zhang et al., 2023a): It dynamically adjusts the rank of incremental matrices to control the parameter budget, with AdaLoRA modules added to all linear layers, similar to LoRA. (3) **Full Fine-tune**: All model parameters, initialized from pre-trained weights and biases, undergo gradient updates during fine-tuning.

Evaluation and Training Setup. We assess language model alignment across two key dimensions: (1) **Language Understanding Ability**: Evaluated using **MMLU** (Hendrycks et al., 2021) for specialized knowledge and **Hellaswag** (Zellers et al., 2019) for commonsense reasoning. (2) **Conversational Ability**: Measured using **MT-Bench** (Zheng et al., 2023) (multi-turn) and **Vicuna** (Chiang et al., 2023) (single-turn), with responses graded by **GPT-4o**. All evaluations are performed three times, and the average scores are reported. **We conduct hyperparameter searches for LoRA and full fine-tuning to establish strong baselines.**

Targeted Performance. (1) **Language Understanding Ability**: Recent research (Du et al., 2020; Sun et al., 2021; Dubey et al., 2024) suggests that the learning of language understanding tasks essentially occurs during the pre-training phase of the base model. Therefore, significant performance improvements in language understanding tasks (i.e., MMLU, Hellaswag) after alignment are not expected. However, *it is crucial to ensure the model retains the learned knowledge during alignment*. (2) **Conversational Ability**: Without alignment, the pre-train model's conversational ability is poor. For example, LLAMA 2-7B often produces incorrect or irrelevant responses on the Vicuna dataset. However, *its conversational ability can be significantly improved through the alignment process*.

For all experiments, we follow fine-tuning hyperparameters: we use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and weight decay of 0.1. The scheduler employed is a cosine scheduler with a warmup ratio of 0.01. For LoRA baselines, we set the hyperparameter rank r as 32.

B.1 No Robots Dataset

We do a hyperparameter search for LoRA over the following variables: learning rate $\{0.001, 0.002, 0.0005, 0.0002, 0.0001\}$, training epochs $\{2, 3, 4, 5\}$. We do hyperparameter search for full fine-tuning over the following variables: learning rate $\{1e-4, 2e-5, 1e-5, 5e-6, 2e-6\}$, training epochs $\{2, 3, 4, 5\}$.

LLAMA 2-7B. Both LoRA and AdaLoRA use a dropout rate of 0.1 and a learning rate of 0.001. The number of training epochs is 3. For full fine-tuning, the learning rate is set to 0.00001, with the number of training epochs also being 3. The training parameters for ILA are consistent with those of the baselines.

Mistral-7B. For LoRA and AdaLorA, we set the dropout rate as 0.1. The learning is 0.0002. The number of training epochs is 2. For full fine-tuning, the learning rate is set as 0.000002 and the number of training epochs is 2. The training parameters of ILA are the same as the baselines.

B.2 LIMA Dataset

We do a hyperparameter search for LoRA over the following variables: learning rate $\{0.001, 0.002, 0.0005, 0.0002, 0.0001\}$, training epochs $\{5, 10, 15, 20\}$. We do hyperparameter search for full fine-tuning over the following variables: learning rate $\{1e-4, 2e-5, 1e-5, 5e-6, 2e-6\}$, training epochs $\{5, 10, 15, 20\}$.

LLAMA 2-7B . For LoRA and AdaLorA, we set the dropout rate as 0.1. The learning is 0.001. The number of training epochs is 20. For full fine-tuning, the learning rate is set as 0.00001 and the number of training epochs is 5. The training parameters of ILA are the same as the baselines.

Mistral-7B. For LoRA and AdaLorA, we set the dropout rate as 0.1. The learning is 0.0002. The number of training epochs is 5. For full fine-tuning, the learning rate is set as 0.000005 and the number of training epochs is 5. The training parameters of ILA are the same as the baselines.

B.3 Alpaca-GPT Dataset.

We do a hyperparameter search for LoRA over the following variables: learning rate $\{0.001, 0.002, 0.0005, 0.0002, 0.0001\}$, training epochs $\{0.5, 1, 1.5, 2, 3\}$. We do hyperparameter search for full fine-tuning over the following variables: learning rate $\{1e-4, 2e-5, 1e-5, 5e-6, 2e-6\}$, training epochs $\{0.5, 1, 1.5, 2, 3\}$.

LLAMA 2-7B . For LoRA and AdaLorA, we set the dropout rate as 0.1. The learning is 0.0002. The number of training epochs is 1.5. For full fine-tuning, the learning rate is set as 0.000002 and the number of training epochs is 0.5. The training parameters of ILA are the same as the baselines.

Mistral-7B. For LoRA and AdaLorA, we set the dropout rate as 0.1. The learning is 0.0002. The number of training epochs is 5. For full fine-tuning, the learning rate is set as 0.000002 and the number of training epochs is 0.5. The training parameters of ILA are the same as the baselines.

C Additional Experiments

C.1 GPU Memory Usage Analysis

We provide an overview of the GPU memory usage for different fine-tuning strategies, as shown in Table 15. The table demonstrates the GPU memory usage and average training time per iteration for various fine-tuning approaches, including LoRA, QLoRA, full fine-tune, and their modified versions where only 30% of the important layers identified by ILA are fine-tuned. Both LoRA and QLoRA show substantial reductions in memory usage when restricted to tuning only 30% of important layers, compared to the full-layer fine-tuning approaches. These results indicate that selectively fine-tuning a small set of critical layers is highly effective in reducing GPU memory consumption, particularly for efficient methods like QLoRA. This suggests that targeted fine-tuning can enhance computational efficiency while preserving model performance, which is especially beneficial when scaling large language models with limited hardware resources.

C.2 Cross-dataset Evaluation of Layer Importance

As shown in Table 2, different datasets reveal subtle variations in the layers identified as important. This suggests that layers consistently deemed unimportant across multiple datasets are likely genuinely non-essential. To validate this, we intersect the top-K least

Table 15: GPU memory usage for LoRA, QLoRA, Full Fine-tune and LoRA/QLoRA/Full Fine-tune with only 30% of important layers fine-tuned. Batch size is set to 2, and the maximum token length is 1024. Percentages in parentheses indicate the proportion of linear layers fine-tuned.

	GPU Memory Usage (MiB)	**Training time (ms)**
Full Fine-tune (100%)	81276	396
Full Fine-tune w/ ILA (30%)	33458	304
LoRA (100%)	32752	403
LoRA w/ ILA (30%)	28586	359
QLoRA (100%)	26238	523
QLoRA w/ ILA (30%)	17912	423

Table 16: Results of fine-tuning Mistral-7B-v0.1 on the LIMA dataset using ILA to identify important layers from various datasets. **Dataset (Imp. Layers)** indicates the datasets utilized to search for the important layers. **Intersection** represents freezing the layers that are the intersection of the top-K least important layers found from the LIMA, No Robots, and Alpaca GPT4 datasets. Evaluated using MMLU (5-shot), Hellaswag (0-shot), **GPT-4 scores** on Vicuna prompts, and MT-Bench prompts.

Dataset (Imp. Layers)	**Dataset (Fine-tune)**	**Language Understanding**		**Conversational Ability**	
		MMLU ↑	**Hellaswag** ↑	**Vicuna** ↑	**MT-Bench** ↑
LIMA	LIMA	**61.82**	65.48	6.99	5.38
No Robots	LIMA	61.52	65.51	6.92	5.34
Alpaca-GPT4	LIMA	61.23	65.20	7.03	5.21
Intersection	LIMA	61.49	**65.62**	**7.06**	**5.44**

important layers identified from three datasets (LIMA, No Robots, and Alpaca-GPT4) to derive a set of universally non-critical layers. The results are presented in Table 16.

Our analysis reveals that a holistic consideration of layer importance across multiple datasets yields superior results compared to dataset-specific approaches. For instance, identifying important layers within the LIMA dataset and fine-tuning on the No Robots dataset is less effective than an integrated approach. Similarly, finding important layers and fine-tuning exclusively on the No Robots dataset do not perform as well as the comprehensive method. This suggests that a cross-dataset evaluation of layer importance can lead to more robust and effective fine-tuning strategies.

C.3 Cross-model Transfer of Layer Importance Rankings

We evaluated the transferability of layer importance rankings across different models, focusing on the Jaccard similarity of the top 75% important layers. The results for various models are shown in Table 10. As seen, the Jaccard similarity between different architectures is approximately **0.70**, suggesting a moderate overlap in the layers identified as important.

In our subsequent experiment, we used LLAMA 2-7B's layer importance rankings from the No Robots dataset to fine-tune Mistral-7B-v0.1, as shown in Table 11. The results demonstrate that while cross-model transfer offers performance improvements, fine-tuning using rankings from the same model architecture yields the best results.

These findings suggest that cross-model transfer of layer importance rankings is possible, though less effective than using rankings from the same architecture. Fine-tuning the top 75% of layers based on cross-model transfer shows some improvement, while fine-tuning only the top 30% achieves comparable performance.

C.4 Impact of Initial Layer Importance Scores

We evaluated the effect of different initial layer importance scores on the consistency of identified important layers. The scores were initialized to $s_0 = 4.0, 2.0, 1.0$. The consistency

was measured using the Jaccard similarity of the top 75% important layers identified during fine-tuning of LLama 2-7B on the LIMA dataset. The results are shown in Table 17. These results show that the method is stable with respect to initialization. While the choice of s_t can influence the optimization trajectory, it does not significantly impact the convergence or final importance rankings.

Table 17: The Jaccard similarities of top 75% important layers identified during fine-tuning of LLAMA 2-7B on the LIMA dataset with varying initial scores.

Initial Scores	**4.0**	**2.0**	**1.0**
4.0	-	-	-
2.0	0.83	-	-
1.0	0.78	0.88	-

C.5 Additional Experiments on Model Scalability

To assess whether freezing unimportant layers continues to enhance model performance at a larger scale, we conducted additional experiments on LLAMA 2-13B. Specifically, we fine-tuned LLAMA 2-13B using the No Robots and LIMA datasets, with results compared against LoRA presented in the table below. The experimental outcomes demonstrate that our method maintains strong performance on LLAMA 2-13B. Despite the increased model size, the underlying architectural similarities suggest that our approach remains effective and scalable, likely extending its benefits to even larger models.

We also carried out further experiments on **LIMA** and **Alpaca-GPT4** using **LLAMA 2-7B**, **Mistral-7B-v0.1** and **Llama 3.1-8B** to evaluate the adaptability of our approach across different model architectures. Consistently, our method outperformed LoRA while requiring fewer layers to be fine-tuned. These findings further validate the robustness and scalability of our approach, showing its capability to effectively enhance performance across various model sizes and architectural variations.

Table 18: Fine-tuning results of LLAMA 2-13B on the LIMA and No Robots datasets. Evaluated using MMLU (5-shot), Hellaswag (0-shot), **GPT-4o scores** on Vicuna prompts, and MT-Bench prompts. Cells highlighted in grey indicate that ILA has improved the performance of the base model.

Datasets	**Methods**	**Language Understanding**		**Conversational Ability**	
		MMLU ↑	**Hellaswag** ↑	**Vicuna** ↑	**MT-Bench** ↑
LIMA	LoRA	53.85	63.08	6.16	3.79
	LoRA w/ ILA	54.33	62.04	6.25	3.91
No Robots	LoRA	54.08	61.73	5.72	4.24
	LoRA w/ ILA	54.45	61.13	5.88	4.37

D ILA for LLM Reasoning

Datasets. (1) **LIMO (Ye et al., 2025):** This dataset comprises 817 carefully selected problems drawn from an initial pool of tens of millions. The final selection meets strict quality standards and covers a broad range of mathematical reasoning tasks. High-quality solutions are provided by both human experts and AI systems like DeepSeek R1 (Guo et al., 2025). (2) **s1.1 (Muennighoff et al., 2025):** This dataset includes 1,000 questions paired with reasoning traces, curated based on three rigorously validated criteria: difficulty, diversity, and quality. The chain-of-thought solutions are generated by DeepSeek R1 (Guo et al., 2025).

Evaluation. (1) AIME24: This set contains 30 problems from the 2024 American Invitational Mathematics Examination (AIME), administered on January 31 and February 1, 2024. (2)

Table 19: Comparative evaluation of LLAMA 2-7B, Mistral-7B-v0.1, and Llama 3.1-8B models fine-tuned on the LIMA Dataset. Evaluated using MMLU (5-shot), Hellaswag (0-shot), **GPT-4o scores** on Vicuna prompts, and MT-Bench prompts. **The evaluations are performed three times, and the average scores are reported.** Cells highlighted in grey indicate that ILA has enhanced the performance of the base model. The best result is marked in bold.

Models	Methods	Language Understanding		Conversational Ability	
		MMLU ↑	Hellaswag ↑	Vicuna ↑	MT-Bench ↑
LLAMA 2-7B	AdaLoRA	44.21	59.85	5.22	3.51
	Full Fine-tune	**46.36**	62.06	5.83	3.71
	Full Fine-tune w/ ILA	46.32	**62.18**	**5.98**	**3.85**
	LoRA	43.18	54.52	5.43	3.45
	LoRA w/ ILA	44.13	54.55	5.62	3.72
Mistral-7B-v0.1	AdaLoRA	**62.40**	61.52	6.64	4.49
	Full Fine-tune	60.11	63.76	6.88	4.63
	Full Fine-tune w/ ILA	61.01	64.01	6.95	**4.77**
	LoRA	60.83	65.42	6.70	4.58
	LoRA w/ ILA	61.52	**65.51**	**6.98**	4.69
Llama 3.1-8B	AdaLoRA	63.55	62.65	6.50	4.73
	Full Fine-tune	64.31	65.64	7.09	5.12
	Full Fine-tune w/ ILA	**64.73**	**65.98**	**7.17**	**5.23**
	LoRA	62.33	62.92	6.57	4.79
	LoRA w/ ILA	63.31	63.01	6.61	4.93

Table 20: Comparative evaluation of LLAMA 2-7B, Mistral-7B-v0.1, and Llama 3.1-8B models fine-tuned on the Alpaca-GPT4 Dataset. Evaluated using MMLU (5-shot), Hellaswag (0-shot), **GPT-4o scores** on Vicuna prompts, and MT-Bench prompts. **The evaluations are performed three times, and the average scores are reported.** Cells highlighted in grey indicate that ILA has enhanced the performance of the base model. The best result is marked in bold.

Models	Methods	Language Understanding		Conversational Ability	
		MMLU ↑	Hellaswag ↑	Vicuna ↑	MT-Bench ↑
llama-7B	AdaLoRA	46.13	57.85	6.89	3.78
	Full Fine-tune	45.91	57.73	6.78	3.72
	Full Fine-tune w/ ILA	**46.23**	57.67	6.99	3.85
	LoRA	43.66	**58.49**	6.96	3.80
	LoRA w/ ILA	44.69	58.22	7.17	3.99
Mistral-7B-v0.1	AdaLoRA	**62.48**	62.08	7.25	4.77
	Full Fine-tune	60.56	62.80	7.19	4.78
	Full Fine-tune w/ ILA	60.88	**62.91**	7.35	4.91
	LoRA	61.82	62.70	7.23	4.89
	LoRA w/ ILA	62.14	62.80	7.33	5.02
Llama 3.1-8B	AdaLoRA	**65.82**	61.02	7.48	5.39
	Full Fine-tune	63.58	61.58	7.33	5.32
	Full Fine-tune w/ ILA	64.61	61.74	7.57	5.42
	LoRA	65.40	61.72	7.65	5.43
	LoRA w/ ILA	65.76	**61.81**	**7.79**	**5.55**

MATH500 (Hendrycks et al., 2021): A benchmark consisting of competition-level math problems spanning a range of difficulties.

Consistency in Layer Importance Across Datasets. We applied our proposed ILA algorithm to identify layer importance rankings on both the LIMO and s1.1 datasets using the Qwen2.5-7B-Instruct model (Yang et al., 2024). The resulting rankings showed strong consistency, with a Jaccard similarity of ***0.86*** (see Fig.3), suggesting that LLMs tend to acquire similar reasoning-related knowledge across datasets. We hypothesize that much of this knowledge is already learned during pretraining—as also suggested by LIMO (Ye et al., 2025)—and that fine-tuning primarily serves to activate the model's latent reasoning abilities.

Based on the identified importance rankings, we further conducted experiments on LIMO by freezing approximately 25% of the least important layers. As shown in Table 14, fine-tuning only the top 75% most important layers led to a slight improvement in performance, indicating that selective tuning can help enhance the model's reasoning capabilities.