DR-LoRA: Dynamic Rank LoRA for Fine-Tuning Mixture-of-Experts Models

Guanzhi Deng[1,2*] **Bo Li**[3*] **Ronghao Chen**[4*] **Xiujin Liu**[5] **Zhuo Han**[1,2]
Huacan Wang[4] **Lijie Wen**[3†] **Linqi Song**[1,2†]

[1] City University of Hong Kong
[2] City University of Hong Kong Shenzhen Research Institute
[3] Tsinghua University [4] QuantaAlpha [5] University of Michigan

Abstract

Mixture-of-Experts (MoE) has become a prominent paradigm for scaling Large Language Models (LLMs). Parameter-efficient fine-tuning methods, such as LoRA, are widely adopted to adapt pretrained MoE LLMs to downstream tasks. However, existing approaches typically assign identical LoRA ranks to all expert modules, ignoring the heterogeneous specialization of pretrained experts. This uniform allocation leads to a resource mismatch: task-relevant experts are under-provisioned, while less relevant ones receive redundant parameters. To address this, we propose **DR-LoRA**, a **D**ynamic **R**ank **LoRA** framework for fine-tuning pretrained MoE models. Specifically, DR-LoRA initializes all expert LoRA modules with a small active rank and uses an expert saliency score, which combines routing frequency and gradient-based rank importance, to identify which experts would benefit most from additional capacity. It then periodically expands the active ranks of the task-critical expert LoRA, progressively constructing a heterogeneous rank distribution tailored to the target task. Experiments on three MoE models across six tasks show that DR-LoRA consistently outperforms LoRA and other strong baselines, demonstrating that task-adaptive heterogeneous rank allocation is an effective strategy to improve active capacity utilization in MoE fine-tuning.

Code: https://github.com/gz-d/dr-lora

1 Introduction

Mixture-of-Experts (MoE) has become a prominent paradigm for scaling Large Language Models (LLMs) (Yang et al., 2025; Jiang et al., 2024; Liu et al., 2024a). By activating only a subset of experts for each input token, MoE substantially increases model capacity without proportionally increasing per-token computation, demonstrating impressive capabilities across a wide range of tasks (Shazeer et al., 2017; Li et al., 2026b; Team et al., 2025; Li et al., 2026a). With the widespread adoption of MoE LLMs, efficiently adapting them to specific downstream tasks has become a significant challenge.

Parameter-Efficient Fine-Tuning (PEFT), particularly LoRA (Hu et al., 2022), is a primary approach to address this challenge. However, when applied to MoE models, existing methods typically assign a LoRA rank to all experts (Li et al., 2024; Liu et al., 2024b; Dou et al., 2024; Gao et al., 2025; Liu et al., 2024d), implicitly assuming that all experts have similar adaptation demands. This assumption overlooks the heterogeneous activation patterns and functional specialization that experts develop during pretraining (Wang et al., 2025; Fedus et al., 2022; Jiang et al., 2024; Dai et al., 2024). Uniform rank allocation thus leads to **Expert-Level Capacity Mismatch**: high-frequency experts lack sufficient parameters to fully adapt, while low-frequency experts occupy capacity that contributes little to task performance.

Previous work provides partial solutions but does not fully address the capacity mismatch problem. For example, PERFT-R (Liu et al., 2024d) introduces routable adaptation modules

*Equal Contribution. †Corresponding Authors.

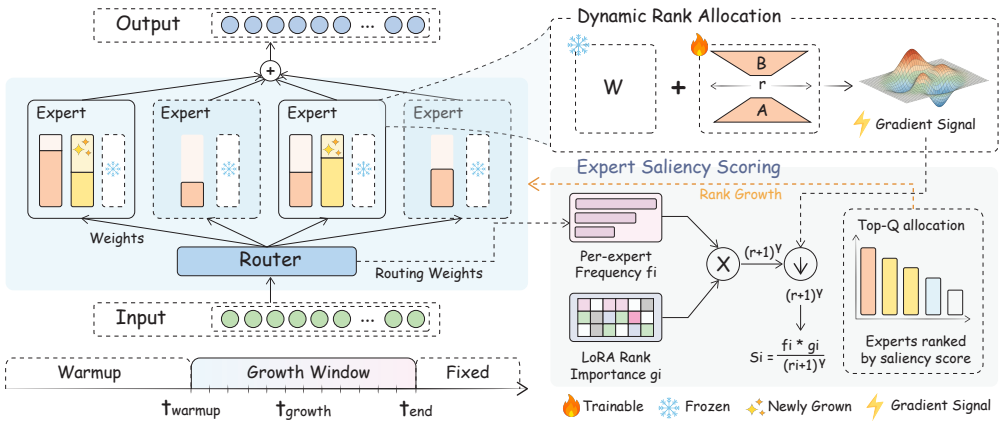


Figure 1: Overview of DR-LoRA. Pre-trained expert weights are frozen; each expert's LoRA module starts at a small initial rank and grows dynamically during training. Expert Saliency Scoring integrates routing frequency and rank importance to quantify each expert's demand for capacity expansion. Dynamic Rank Allocation periodically expands active ranks for high-saliency experts, progressively forming a task-adaptive heterogeneous rank structure.

to improve MoE fine-tuning but uses uniform rank configurations for all experts, without differentiating based on actual usage. AdaLoRA (Zhang et al., 2023) achieves dynamic rank allocation through importance-driven pruning but is designed for dense models. It evaluates each LoRA module independently based on gradient signals, without leveraging the routing structure that distinguishes frequently used experts from those rarely activated. Additionally, its top-down pruning approach is structurally mismatched with MoE's sparse routing. In such sparse settings, low-frequency experts receive only sparse and noisy gradient signals during training, making their importance estimates unreliable and the resulting pruning decisions vulnerable to noise.

These observations suggest two key requirements for effective MoE fine-tuning: **(I)** Rank allocation should reflect the heterogeneous usage patterns of experts on the target task. **(II)** The capacity adjustment process must accommodate the sparse and progressive nature of MoE training signals. To address this, we propose **DR-LoRA**, a growth-based dynamic rank allocation framework for MoE fine-tuning. As shown in Figure 1, DR-LoRA achieves task-adaptive heterogeneity through two core components. First, Expert Saliency Scoring combines routing frequency and gradient-based rank importance to score experts for rank growth. This module prioritizes experts that are both frequently routed and still actively learning. Second, Dynamic Rank Allocation initializes all experts with a small active rank and periodically expands the ranks of high-score experts, avoiding the unreliable early pruning decisions that can arise in top-down methods under sparse routing. We evaluated DR-LoRA on three MoE models across six tasks. The results show that it consistently outperforms LoRA and other strong PEFT baselines.

The contributions of this work are summarized as follows: **(I)** We identify an expert-level capacity mismatch in MoE fine-tuning, where LoRA rank allocation overlooks the heterogeneous expert usage caused by sparse routing, under-provisioning high-frequency experts while wasting capacity on low-frequency experts. **(II)** We propose DR-LoRA, a growth-based dynamic rank allocation framework for effective MoE fine-tuning by combining routing frequency and gradient-based rank importance, and a growth-based allocation mechanism for sparse routing. **(III)** We validate the effectiveness of our method on three MoE models. Further analysis confirms that DR-LoRA effectively concentrates the trainable parameter budget on task-critical experts, resulting in a heterogeneous rank distribution.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT) aims to efficiently adapt LLMs by introducing a small number of trainable parameters. LoRA (Hu et al., 2022) and its subsequent improve-

ments (Liu et al., 2024c; Hayou et al., 2024) are primarily designed for dense models and do not account for the heterogeneous expert activation patterns introduced by MoE routing. With the development of MoE models, several works begin to explore the application of LoRA to MoE models (Tang et al., 2026; Pirchert et al., 2026). Typical methods directly inject LoRA into each expert and use a uniform rank allocation strategy. However, these methods implicitly assume that all experts require similar fine-tuning capacity, overlooking the task-dependent imbalance in expert activation induced by routing (Dai et al., 2024; Wang et al., 2024). (Dai et al., 2024; Wang et al., 2024). FlexMoRE (Pirchert et al., 2026) investigates rank-heterogeneous experts in federatedly trained MoE and finds that reasoning-heavy tasks benefit from higher expert ranks, providing independent evidence for capacity mismatch. Other works introduce MoE structures as plugins in dense models, such as MoLA (Gao et al., 2025), MiLoRA (Li et al., 2024), CoMoE (Feng et al., 2025), and LoRAMoE (Dou et al., 2024). These methods construct routing mechanisms from scratch to expand the model capacity, which differs fundamentally from fine-tuning of pretrained MoE experts, as studied in this paper. In recent research, PERFT (Liu et al., 2024d) introduces a unified PEFT framework for MoEs, enhancing model expressiveness by building routable adaptation modules (e.g., PERFT-R). Although these methods make use of MoE routing, their primary focus is on the design and integration of adaptation modules, rather than on fine-grained capacity reallocation across the original experts. AdaLoRA (Zhang et al., 2023) achieves adaptive rank allocation through singular value pruning. This method is effective in dense models, and budgets are allocated using module-level gradient signals. However, its importance evaluation assumes that all modules receive sufficient gradient signals, and its top-down pruning process requires all modules to begin training at a high rank. Under the sparse routing of MoEs, low-frequency experts accumulate only sparse and noisy gradients, making their importance estimates unreliable.

Unlike previous works, DR-LoRA uses both routing frequency and gradient-based rank importance to guide rank allocation. By adopting a growth-based mechanism that matches the gradual emergence of expert-specific learning signals under sparse routing, it enables fine-grained dynamic capacity allocation at the expert level.

3 Methodology

In this section, we present the DR-LoRA framework, which dynamically allocates expert LoRA ranks during fine-tuning. The architecture overview is shown in Figure 1. We first introduce the problem setting and the motivation for the rank allocation rule (§3.1), then describe Expert Saliency Scoring (§3.2) and Dynamic Rank Allocation (§3.3).

3.1 Preliminaries and Problem Formulation

We consider a pretrained MoE LLM with L transformer layers, each containing N experts and a router G_ℓ. For an input token x, the router computes routing weights and activates the top-k experts. We denote the post-top-k routing weight as $z_{\ell,i}(x)$ (zero when expert i is not selected), so that $\text{MoE}_\ell(x) = \sum_i z_{\ell,i}(x) \cdot E_{\ell,i}(x)$. LoRA adapts each expert's feed-forward projections via low-rank injection: $\mathbf{W}' = \mathbf{W} + \mathbf{BA}$. Each expert contains $P = 2$ adapted projections (`up_proj` and `down_proj`), sharing a single expert-level active rank $r_{\ell,i}^{(t)}$ at training step t. Our goal is to construct a heterogeneous rank distribution such that the total active ranks at the end of training match LoRA: $\sum_{\ell,i} P \cdot r_{\ell,i}^{(t_{\text{end}})} = L \times N \times P \times r_{\text{target}}$. To support dynamic growth, each expert pre-allocates physical parameter space up to $r_{\max} = 2\, r_{\text{target}}$, but only the active dimensions are used in the forward pass.

The key question is: *which expert should receive the next rank dimension to achieve the largest loss reduction?* In MoE models, expert gradients are inherently modulated by their routing weights, and an unselected expert receives no gradient signal. Using a first-order approximation and a mean-field factorization (detailed in Appendix A), the expected benefit of

expanding expert i's capacity can be approximated as:

$$\Delta\mathcal{L}_{\ell,i} \propto \underbrace{\mathbb{E}[z_{\ell,i}]}_{f_{\ell,i}\text{: routing frequency}} \cdot \underbrace{\mathbb{E}[q_{\ell,i} \mid z_{\ell,i} > 0]}_{g_{\ell,i}\text{: learning intensity when activated}}. \tag{1}$$

Here, $q_{\ell,i}(x)$ denotes the effective local gradient intensity of expert i when activated. We treat this factorization as a theoretically motivated heuristic, of which the practical value lies in capturing two distinct and complementary aspects of expert demand (Appendix A discusses the validity of the underlying approximation). The two factors have clear semantics: $f_{\ell,i}$ measures the frequency with which the expert is used by the task data, and $g_{\ell,i}$ captures its average gradient activity when activated. Intuitively, an expert should be expanded only when it is frequently routed and still exhibits strong learning signals.

3.2 Expert Saliency Scoring

Based on the analysis in §3.1, we construct Expert Saliency Scoring from two complementary signals: routing frequency and rank importance.

Routing frequency (f). Let $B^{(t)}$ denote the set of tokens at training step t. We compute the step-level mean post-top-k routing weight and track its exponential moving average (EMA):

$$\bar{z}_{\ell,i}^{(t)} = \frac{1}{|B^{(t)}|} \sum_{x \in B^{(t)}} z_{\ell,i}(x), \qquad f_{\ell,i}^{(t)} = \beta\, f_{\ell,i}^{(t-1)} + (1-\beta)\, \bar{z}_{\ell,i}^{(t)}, \tag{2}$$

where $\beta \in [0, 1)$ is the decay coefficient.

Rank importance (g). Routing frequency captures data relevance but does not guarantee that the expert is actively learning (e.g., it may have already converged). Therefore, we also measure learning intensity through gradient-weight products (Zhang et al., 2023). For the j-th active rank dimension, parameterized by $\mathbf{a}_j = \mathbf{A}_{\ell,i}[j,:]$ and $\mathbf{b}_j = \mathbf{B}_{\ell,i}[:,j]$, its sensitivity at step t with respect to training loss $\mathcal{L}^{(t)}$ is:

$$s_{\ell,i,j}^{(t)} = \left\| \frac{\partial \mathcal{L}^{(t)}}{\partial \mathbf{a}_j} \odot \mathbf{a}_j \right\|_1 \cdot \left\| \frac{\partial \mathcal{L}^{(t)}}{\partial \mathbf{b}_j} \odot \mathbf{b}_j \right\|_1. \tag{3}$$

We use a multiplicative form because a rank dimension contributes meaningfully only when both its A-side and B-side carry nontrivial gradient-weight signals. Each dimension's sensitivity is smoothed using an EMA, then aggregated to the expert level by averaging over active dimensions. Let $\mathbf{m}_{\ell,i}^{(t)} \in \{0,1\}^{r_{\max}}$ be the binary mask that tracks active rank dimensions, with its j-th entry denoted $m_{\ell,i,j}^{(t)}$. The expert-level rank importance is:

$$g_{\ell,i,j}^{(t)} = \beta\, g_{\ell,i,j}^{(t-1)} + (1-\beta)\, s_{\ell,i,j}^{(t)}, \qquad g_{\ell,i}^{(t)} = \frac{1}{r_{\ell,i}^{(t)}} \sum_{j:\, m_{\ell,i,j}^{(t)}=1} g_{\ell,i,j}^{(t)}. \tag{4}$$

The average of $r_{\ell,i}^{(t)}$ avoids systematically favoring experts that already have more active ranks, making $g_{\ell,i}^{(t)}$ a rank-normalized measure of learning intensity. Appendix A.2 provides a further justification for the multiplicative form and the choice of averaging over alternative aggregation strategies.

Saliency score. We define the expert saliency score:

$$S_{\ell,i}^{(t)} = \frac{f_{\ell,i}^{(t)} \cdot g_{\ell,i}^{(t)}}{(r_{\ell,i}^{(t)} + 1)^{\gamma}}. \tag{5}$$

The numerator $f \cdot g$ serves as a practical proxy for the relative benefit of expanding expert i. The denominator assigns decreasing growth priority to experts that already hold higher

ranks: an expert must exhibit stronger signals to continue receiving growth, preventing a small number of experts from monopolizing the parameter budget. γ controls this effect: $\gamma = 0$ removes it entirely, causing performance degradation, while excessively large γ suppresses meaningful differentiation (see §5.4).

3.3 Dynamic Rank Allocation

Using continuously updated saliency scores, DR-LoRA progressively grows each layer's active ranks from r_{init} toward the target training budget.

Initialization. Each expert pre-allocates LoRA matrices up to r_{max} dimensions but activates only r_{init} at the start. The binary mask $\mathbf{m}_{\ell,i}^{(t)}$ introduced in §3.2 tracks active dimensions, and the forward pass uses only the active subspace: $\mathbf{W}' = \mathbf{W} + \mathbf{B}[:, \mathbf{m}_{\ell,i}^{(t)}] \cdot \mathbf{A}[\mathbf{m}_{\ell,i}^{(t)}, :]$.

Growth schedule. Rank grows every T_{grow} steps within a growth window T_{window}. Let N_{grow} denote the number of growth events; at each event, a fixed per-layer quota $Q = \lceil N \times (r_{\text{target}} - r_{\text{init}})/N_{\text{grow}} \rceil$ of new ranks is distributed, reaching the target active-rank budget by the end of T_{window}. See Appendix B.2.2 for detailed schedule parameters.

Per-layer greedy allocation. At each growth event, DR-LoRA processes each layer independently. All N experts in layer ℓ are sorted by $S_{\ell,i}^{(t)}$ in descending order, and the quota Q is greedily distributed to the top-scoring experts. Per-layer allocation prevents layers with systematically larger saliency scores in practice from absorbing a disproportionate share of the budget. The number of new ranks assigned to each expert is capped:

$$n_{\text{grow}} = \min\Big(\lfloor (r_{\text{max}} - r_{\text{init}}) \cdot p_{\text{grow}} \rfloor,\ r_{\text{max}} - r_{\ell,i}^{(t)},\ Q_{\text{remain}} \Big), \tag{6}$$

here p_{grow} limits the maximum growth fraction per event, ensuring sufficiently gradual growth for the saliency scoring to distinguish experts with a genuinely high-demand from transient fluctuations. n_{grow} is computed for each expert $E_{\ell,i}$ sequentially in descending saliency order, and Q_{remain} denotes the remaining per-layer quota, which decreases as experts are allocated. After each growth event, we reset rank importance scores while preserving routing frequencies, so that newly expanded experts must justify further growth through new gradient signals rather than historical importance.

4 Experiments

4.1 Experimental Setup

Models. We evaluated DR-LoRA in three MoE models with different routing mechanisms and parameter sizes: **OLMoE-1B-7B** (Muennighoff et al., 2025), **Qwen1.5-MoE-A2.7B** (Team, 2024), and **LLaMA-MoE-v1-3.5B (4/16)** (Zhu et al., 2024).

Training and Evaluation. To evaluate DR-LoRA's ability to exploit task-specific expert specialization, we conduct independent fine-tuning on six domains, each paired with standardized benchmarks: **Mathematical Reasoning** (GSM8K (Cobbe et al., 2021)), **Code Generation** (HumanEval (Chen, 2021)), **Instruction Following** (IFEval (Zhou et al., 2023)), **Medical QA** (MedMCQA (Pal et al., 2022)), **Machine Translation** (WMT23 (Kocmi et al., 2023)), and **Legal Understanding** (LEDGAR (Tuggener et al., 2020)). Training datasets and evaluation details are provided in Appendix B.

Baselines. We compare DR-LoRA with the pretrained **Base Model** and several strong PEFT baselines: **LoRA** (Hu et al., 2022), **AdaLoRA** (Zhang et al., 2023), and **PERFT-R** (Liu et al., 2024d). DR-LoRA uses a final average active rank of $r_{\text{target}} = 16$ but pre-allocates trainable

https://github.com/google-research/mt-metrics-eval

Method	#Trainable	Math	Code	Instruct	Medical	Translation	Legal	Avg
		GSM8K	HumanEval	IFEval	MedMCQA	WMT23	LEDGAR	
OLMoE-1B-7B								
Base	–	12.9±0.4	13.6±0.3	15.9±0.8	35.7±0.7	56.6±0.8	62.2±1.2	32.8
LoRA (r=32)	201.3M	25.2±0.2	14.8±0.8	23.3±0.4	40.1±1.0	63.2±1.0	79.4±0.8	41.0
AdaLoRA	201.3M	26.3±0.3	15.3±0.8	24.1±0.5	41.5±1.0	63.8±1.0	80.1±0.9	41.9
PERFT-R	201.3M	25.9±0.3	15.5±0.8	24.3±0.5	41.0±0.9	64.0±1.0	79.6±0.9	41.7
DR-LoRA (Ours)	201.3M	**28.4**±1.0	**16.7**±0.4	**26.7**±1.0	**43.9**±0.8	**66.5**±1.4	**82.1**±1.1	**44.1**
Qwen1.5-MoE-A2.7B								
Base	–	61.7±1.1	35.0±0.8	23.5±0.6	43.2±1.0	72.6±1.3	83.2±1.0	53.2
LoRA (r=32)	318.5M	64.5±1.1	43.9±0.8	32.0±0.3	47.4±1.2	78.0±1.2	92.6±0.7	59.7
AdaLoRA	318.5M	65.4±1.0	45.2±0.7	33.0±0.4	48.0±1.1	79.3±1.1	93.1±0.8	60.7
PERFT-R	318.5M	65.6±1.0	44.8±0.8	33.3±0.4	48.3±1.2	79.1±1.1	93.4±0.8	60.8
DR-LoRA (Ours)	318.5M	**67.2**±1.1	**46.5**±0.6	**35.1**±0.2	**50.1**±0.8	**80.7**±0.6	**94.8**±1.0	**62.4**
LLaMA-MoE-3.5B								
Base	–	3.8±0.8	1.3±0.5	14.9±0.5	25.8±0.7	55.0±1.0	29.6±0.8	21.7
LoRA (r=32)	156.8M	12.3±0.8	10.2±0.5	19.7±0.5	30.3±0.5	63.2±0.8	79.3±1.3	35.8
AdaLoRA	156.8M	13.6±0.7	10.9±0.6	21.2±0.6	31.1±0.6	64.8±0.9	79.8±1.2	36.9
PERFT-R	156.8M	13.1±0.7	11.1±0.5	20.5±0.6	31.3±0.5	64.1±0.9	80.0±1.2	36.7
DR-LoRA (Ours)	156.8M	**15.7**±0.5	**13.1**±0.6	**23.4**±0.8	**32.9**±0.9	**65.8**±1.1	**81.3**±1.2	**38.7**

Table 1: Task-specific adaptation results on three MoE models across six domains. We report mean accuracy (%) ±std over 3 random seeds. **Bold** indicates the best result and underline indicates the second best.

parameter space up to $r_{\max} = 32$. Accordingly, LoRA ($r = 32$) serves as the primary budget-matched baseline, with the same trainable parameter count and training-time memory footprint as DR-LoRA. All methods use identical training schedules, optimization settings, and data orders.

4.2 Main Results

Table 1 presents the main results of DR-LoRA on three pretrained MoE models. Under matched trainable-parameter budgets, DR-LoRA achieves the best average performance on all three models, reaching 44.1, 62.4, and 38.7 on OLMoE, Qwen1.5-MoE, and LLaMA-MoE, respectively, and outperforming the strongest baseline by 2.2, 1.6, and 1.8 points. Moreover, DR-LoRA outperforms the strongest competing baseline in each of the 18 model–task settings, with a mean improvement of 1.8 points. A paired t-test over the 18 setting-level differences confirms that the aggregate improvement is statistically significant ($p < 0.001$); detailed per-setting comparisons are provided in Appendix C.1.

The comparison with fixed-rank LoRA shows that uniformly assigning the same adaptation capacity to all experts is suboptimal for MoE fine-tuning. Although LoRA ($r = 32$) matches DR-LoRA in trainable-parameter count, DR-LoRA consistently achieves higher performance, indicating the advantage of fine-grained expert-level capacity allocation over a uniform rank configuration. The comparison with AdaLoRA further suggests that, while dynamic rank allocation is beneficial, adjusting the budget solely based on module-level gradient importance may still be insufficient to capture the heterogeneous capacity demands of different MoE experts. AdaLoRA follows a top-down pruning process, starting from a higher-rank initialization and progressively pruning toward the target budget. Under sparse routing, low-frequency experts often receive limited and unstable gradient signals, making their importance estimates more susceptible to noise. In contrast, DR-LoRA starts from a small active rank and expands capacity only after expert-specific routing and learning signals have accumulated, which enables more reliable allocation decisions. Finally, the comparison with PERFT-R suggests that enhancing adaptation through routable adaptation modules alone may not fully address the expert-level capacity mismatch. PERFT-R focuses on the design and routing of additional adaptation modules, whereas DR-LoRA directly reallocates capacity across the original pretrained experts. The consistent advantage of DR-LoRA therefore demonstrates the benefit of directly performing fine-grained dynamic rank allocation over the original experts.

Variant	GSM8K	HumanEval	IFEval	Avg
w/o Routing Freq. (g only)	26.5	15.6	24.5	22.2
w/o Rank Imp. (f only)	27.2	15.3	24.1	22.2
Full DR-LoRA	**28.4**	**16.7**	**26.7**	**23.9**

Table 2: Ablation study of saliency score components on OLMoE.

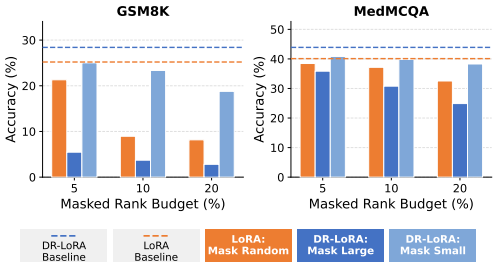
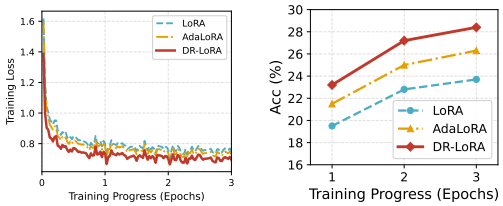


Figure 2: Expert masking on OLMoE. For DR-LoRA, masking large-rank experts causes greater degradation than masking small-rank experts, while uniform-rank LoRA shows a less differentiated pattern.



(a) Training loss (b) Accuracy

Figure 3: Training dynamics of DR-LoRA vs. baselines on OLMoE fine-tuned for GSM8K. DR-LoRA achieves consistently lower loss and higher average accuracy on GSM8K throughout training.

4.3 Ablation Study

We conducted ablation experiments on OLMoE to validate the key design components of DR-LoRA. We report results on three representative benchmarks: GSM8K, HumanEval, and IFEval. Table 2 compares three variants of the saliency score: (1) full DR-LoRA; (2) w/o routing frequency, using only gradient-based rank importance $g_{\ell,i}$; and (3) w/o rank importance, using only routing frequency $f_{\ell,i}$. Full DR-LoRA achieves the best average score of 23.9, outperforming both ablated variants by 1.7 points. These results indicate that routing frequency and rank importance capture complementary aspects of expert demand. Another question is whether the gains of DR-LoRA mainly result from dynamic rank allocation or simply from jointly training the MoE router. To better distinguish these factors, we compare LoRA, AdaLoRA, and DR-LoRA under both *frozen* and *unfrozen* router settings. As shown in Table 3, unfreezing the router improves all three methods, but the advantage of DR-LoRA cannot be attributed to router training alone. Notably, DR-LoRA with a frozen router (Avg 22.5) still outperforms LoRA with an unfrozen router (Avg 21.1). This suggests that expert-level capacity allocation provides substantial gains beyond simply updating the router under a uniform rank configuration. When the router is unfrozen, DR-LoRA achieves further gains, reaching the best overall performance.

Method	Router	GSM8K	HumanEval	IFEval	Avg
LoRA	Frozen	24.1	14.1	22.2	20.1
LoRA	Unfrozen	25.2	14.8	23.3	21.1
AdaLoRA	Frozen	25.0	14.5	22.8	20.8
AdaLoRA	Unfrozen	26.3	15.3	24.1	21.9
DR-LoRA	Frozen	26.9	15.5	25.2	22.5
DR-LoRA	Unfrozen	**28.4**	**16.7**	**26.7**	**23.9**

Table 3: Impact of router training on OLMoE.

5 Analysis

5.1 Masking Analysis of the Learned Rank Allocation

The key motivation of DR-LoRA is that LoRA rank allocation in MoE fine-tuning should not remain uniform across experts. Instead, the learned rank distribution should reflect which experts are more relevant to downstream performance on the target task. To examine whether the learned rank allocation exhibits such a pattern, we perform a masking analysis on OLMoE fine-tuned for GSM8K and MedMCQA. For each fine-tuned model, we partition experts into a large rank group (top 25% by final LoRA rank) and a small rank group (the

Strategy	GSM8K	HumanEval	IFEval	Avg
LoRA	25.2	14.8	23.3	21.1
Random	25.4	15.0	23.6	21.3
Proportional	25.9	15.4	24.8	22.0
Global Greedy	24.8	14.5	22.9	20.7
Per-Layer Greedy	**28.4**	**16.7**	**26.7**	**23.9**

Table 5: Comparison of rank allocation strategies on OLMoE. All variants use the same saliency scores and parameter budget; only the allocation mechanism differs.

γ	GSM8K	HumanEval	IFEval	Avg	Gini
0 (no penalty)	23.1	14.1	24.3	20.5	0.62
0.5	26.2	15.5	26.1	22.6	0.45
0.8	27.6	16.2	26.7	23.5	0.38
1.2 (default)	**28.4**	**16.7**	**26.7**	**23.9**	**0.31**
2.0	27.2	16.3	26.4	23.3	0.22
3.0	26.1	15.4	24.8	22.1	0.14
LoRA (r=32)	25.2	14.8	23.3	21.1	0.00

Table 6: Effect of rank penalty exponent γ on OLMoE. The Gini coefficient measures the final rank concentration across experts.

remaining 75%). We then mask entire expert modules under three masking budgets (5%, 10%, and 20% of the total rank budget per layer), and compare the resulting performance degradation. For the uniform-rank LoRA baseline, we randomly mask experts under the same rank budgets. Figure 2 shows a consistent pattern across both tasks: masking the large-rank expert group in DR-LoRA causes substantially larger degradation than masking the small-rank group. In contrast, masking experts in the uniform-rank LoRA model produce a less differentiated degradation pattern and do not exhibit the same degree of separation. These results suggest that uniform rank allocation leads to a more even distribution of LoRA ranks across experts, whereas DR-LoRA learns a more differentiated rank allocation in which a smaller subset of experts contributes more strongly to downstream performance. This observation is consistent with our motivation that LoRA rank allocation in MoE fine-tuning should better reflect task-specific expert relevance.

5.2 Analysis of Growth-Based Rank Allocation

We analyze the growth-based design of DR-LoRA from two complementary perspectives. First, we compare growth-based and pruning-based dynamic rank allocation. Second, we separate the benefit of progressive growth from that of the final heterogeneous rank structure.

Growth versus pruning. Figure 3 compares the training dynamics of fixed-rank LoRA, pruning-based AdaLoRA, and growth-based DR-LoRA on OLMoE fine-tuned for GSM8K. DR-LoRA converges faster and maintains lower training loss and higher evaluation accuracy throughout training. Although AdaLoRA consistently outperforms fixed-rank LoRA, it remains behind DR-LoRA at all stages. AdaLoRA starts from a high-rank configuration and progressively removes dimensions based on importance estimates. Under sparse routing, low-frequency experts receive limited and unstable gradients, making early pruning decisions susceptible to noise. In contrast, DR-LoRA expands capacity only after sufficient expert-specific routing and learning signals have accumulated, making bottom-up growth better aligned with the progressive emergence of expert importance during MoE fine-tuning.

Progressive growth versus fixed heterogeneous allocation. To separate the contribution of the final rank structure from that of progressive growth, we construct **Static-Final-Rank**. This variant fixes the final per-expert rank distribution discovered by DR-LoRA from initialization and retrains the model without saliency scoring or dynamic growth. As shown in Table 4, Static-Final-Rank improves the average score over LoRA ($r = 32$) from 42.8 to 45.1, showing that the learned heterogeneous allocation is itself beneficial. Full DR-LoRA further improves the average to 46.3, demonstrating that progressive growth contributes an additional 1.2 points beyond the final rank structure. This suggests that the gains arise from both task-adaptive capacity allocation and the progressive process used to construct it.

Method	GSM8K	MedMCQA	WMT23	Avg.
LoRA ($r = 32$)	25.2	40.1	63.2	42.8
Static-Final-Rank	27.6	42.4	65.2	45.1
DR-LoRA	**28.4**	**43.9**	**66.5**	**46.3**

Table 4: Final heterogeneous allocation versus progressive growth on OLMoE.

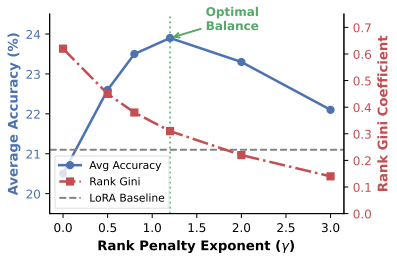
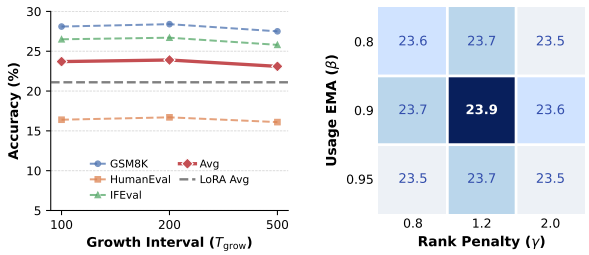


Figure 4: Rank penalty controls the capacity concentration–performance trade-off. Left axis: average accuracy (solid); right axis: rank Gini coefficient (dashed).



(a) Impact of Growth Interval. (b) β-γ Grid Search

Figure 5: Hyperparameter Robustness on OLMoE. **(a)** Performance across different rank growth intervals (T_{grow}). **(b)** A 3×3 grid search over the usage EMA coefficient (β) and rank penalty (γ).

5.3 Allocation Strategy Comparison

We compare five rank allocation strategies on OLMoE under the same total training rank budget, while keeping the saliency score fixed. The compared strategies differ only in how the growth quota is assigned: (I) LoRA: uses a fixed rank for all experts and does not perform dynamic allocation. (II) Random: randomly selects experts for rank growth, without using saliency scores. (III) Proportional: distributes the per-layer quota across all experts in proportion to their normalized saliency scores, so that higher-saliency experts receive more ranks, but lower-saliency experts may still obtain a share of the budget. (IV) Global Greedy: pools the quota across all layers and greedily allocates it to the globally highest-scoring experts, regardless of which layer they belong to. (V) Per-Layer Greedy (ours): allocates quota independently within each layer by greedily assigning ranks to the highest-scoring experts in that layer. Table 5 suggests three main observations. First, saliency-guided allocation is important: Random growth provides only a marginal improvement over LoRA (+0.2 avg), whereas saliency-guided strategies yield substantially larger gains, indicating that the allocation policy itself matters more than merely introducing dynamic growth. Second, concentrated allocation performs better than diluted allocation: Per-Layer Greedy surpasses Proportional by +1.9 avg, suggesting that spreading rank growth across many moderately salient experts is less effective than concentrating capacity on the highest-demand ones. Third, per-layer allocation helps avoid cross-layer imbalance: Global Greedy performs below LoRA (-0.4 avg), which is consistent with the tendency of some layers—often deeper ones—to accumulate systematically higher saliency scores and absorb too much of the budget. Per-Layer Greedy mitigates this issue by enforcing independent budgets within each layer.

5.4 Rank Penalty Analysis

The rank penalty $(r+1)^{-\gamma}$ is introduced to discourage excessive concentration of rank growth in a small number of experts. To study its effect, we vary the penalty exponent γ on OLMoE and report both downstream performance and the Gini coefficient of the final rank distribution. As shown in Table 6, removing the penalty entirely (γ=0) leads to poor performance (20.5 avg), even below LoRA (21.1 avg), and produces the most concentrated rank distribution (Gini $= 0.62$). As γ increases from 0 to 1.2, performance improves steadily while rank concentration decreases. The best result is achieved at γ=1.2, which yields the highest average score (23.9) with a moderate degree of rank inequality (Gini $= 0.31$). When γ becomes too large, performance drops again, and the rank distribution becomes increasingly uniform, suggesting that overly strong regularization suppresses useful differentiation across experts. Figure 4 further visualizes the relationship between penalty strength, rank inequality, and average task performance. Highly concentrated allocations (γ close to 0) are associated with poor performance, while overly uniform allocations (large γ) also reduce performance. Instead, DR-LoRA performs best under a moderate level of rank heterogeneity, where capacity is preferentially assigned to high-demand experts without

collapsing onto only a few of them. This result supports the role of the rank penalty in stabilizing expert-level capacity allocation and preventing both rank monopolization and over-regularization.

5.5 Hyperparameter Robustness

Beyond LoRA, DR-LoRA introduces three additional hyperparameters: the EMA coefficient β, the rank penalty exponent γ, and the growth interval T_{grow}. We evaluate their sensitivity on OLMoE to verify that the performance gains of DR-LoRA are not due to fragile hyperparameters. As shown in Figure 5(a), all tested growth intervals (100, 200, and 500 steps) outperform the LoRA baseline by a clear margin. The default setting of 200 steps achieves the highest average score (23.9), suggesting a favorable trade-off between stable saliency estimation and sufficiently responsive capacity updates. Figure 5(b) shows the 3×3 grid search over β and γ. All configurations consistently outperform the LoRA baseline, and performance varies only within a narrow range. This indicates that DR-LoRA is not particularly sensitive to moderate changes in the saliency-scoring hyperparameters. These results show that DR-LoRA is robust to its main hyperparameters. Therefore we use the same default settings (β=0.9, γ=1.2, T_{grow}=200) across all models and downstream tasks in the main experiments (Table 1), without per-task tuning.

6 Conclusion

In this work, we introduce DR-LoRA, a dynamic rank allocation for LoRA in the context of PEFT for pretrained MoE LLMs. The core idea is to move away from the standard practice of allocating a fixed, uniform rank for all expert LoRA modules during fine-tuning. Instead, DR-LoRA dynamically grows expert LoRA ranks based on a combination of routing frequency and gradient-based rank importance, aiming to match adaptive capacity to each expert's downstream task relevance. A key finding is that growth-based allocation better suits MoE fine-tuning than pruning-based alternatives, as it avoids unreliable importance estimates from sparse gradients in low-frequency experts. The framework is evaluated on three open MoE LLMs across six benchmarks, outperforming uniform-rank baselines and state-of-the-art adaptive methods in terms of accuracy, efficiency, and robustness. Several ablations and diagnostic visualizations support the method's claims and illuminate the dynamics of heterogeneous rank specialization.

Acknowledgements

This work was supported in part by the Research Grants Council of the Hong Kong SAR under Grant GRF 11217823, 11216225, and Collaborative Research Fund C1042-23GF, the National Natural Science Foundation of China under Grant 62371411, InnoHK initiative, the Government of the HKSAR, Laboratory for AI-Powered Financial Technologies, and the National Key Research and Development Program of China (Grant No. 2024YFB3309702).

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. Findings of the 2021 conference on machine translation (WMT21). In Loic Barrault, Ondrej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussa, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Tom Kocmi, Andre Martins, Makoto Morishita, and Christof Monz (eds.), *Proceedings of the Sixth Conference on Machine Translation*, pp. 1–88, Online, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.wmt-1.1/.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 conference on machine translation (WMT19). In Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Marco Turchi, and Karin Verspoor (eds.), *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pp. 1–61, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5301. URL https://aclanthology.org/W19-5301/.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2020 conference on machine translation (WMT20). In Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri (eds.), *Proceedings of the Fifth Conference on Machine Translation*, pp. 1–55, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.wmt-1.1. URL https://aclanthology.org/2020.wmt-1.1/.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. Findings of the 2017 conference on machine translation (WMT17). In Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, and Julia Kreutzer (eds.), *Proceedings of the Second Conference on Machine Translation*, pp. 169–214, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4717. URL https://aclanthology.org/W17-4717/.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. Findings of the 2018 conference on machine translation (WMT18). In Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor (eds.), *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pp. 272–303, Belgium, Brussels, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6401. URL https://aclanthology.org/W18-6401/.

Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca, 2023.

Mark Chen. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, et al. Loramoe: Alleviating world knowledge forgetting in large language models via moe-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1932–1945, 2024.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Jinyuan Feng, ChaoPeng Wei, Tenghai Qiu, Tianyi Hu, and Zhiqiang Pu. CoMoE: Contrastive representation for mixture-of-experts in parameter-efficient fine-tuning. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 7533–7551, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.398. URL https://aclanthology.org/2025.findings-emnlp.398/.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Ruibo Liu, Baochen Sun, Yawen Zhang, Daiyi Peng, Xiaoyuan Guo, and VS Subrahmanian. Mola: Moe lora with layer-wise expert allocation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5097–5112, 2025.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

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. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Tom Kocmi, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Thamme Gowda, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Rebecca Knowles, Philipp Koehn, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Michal Novák, Martin Popel, and Maja Popović. Findings of the 2022 conference on machine translation (WMT22). In Philipp Koehn, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri (eds.), *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pp. 1–45, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.wmt-1.1. URL https://aclanthology.org/2022.wmt-1.1/.

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, Masaaki Nagata, Toshiaki Nakazawa, Martin Popel, Maja Popović, Mariya Shmatova, and Jun Suzuki. Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In Philipp Koehn, Barry Haddow, Tom Kocmi, and Christof Monz (eds.), *Proceedings of the Eighth Conference on Machine Translation*, pp. 1–42, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.wmt-1.1. URL https://aclanthology.org/2023.wmt-1.1/.

Bo Li, Tianyu Dong, Shaolin Zhu, and Deyi Xiong. Mix-moe: Improving multilingual machine translation of large language models through mixed moes. *IEEE Transactions on Audio, Speech and Language Processing*, 2026a.

Bo Li, Chuan Wu, and Shaolin Zhu. MACS: Modality-aware capacity scaling for efficient multimodal MoE inference. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22134–22148, San Diego, California, United States, July 2026b. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1012. URL https://aclanthology.org/2026.acl-long.1012/.

Dengchun Li, Yingzi Ma, Naizheng Wang, Zhengmao Ye, Zhiyuan Cheng, Yinghao Tang, Yan Zhang, Lei Duan, Jie Zuo, Cal Yang, et al. Mixlora: Enhancing large language models fine-tuning with lora-based mixture of experts. *arXiv preprint arXiv:2404.15159*, 2024.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1104–1114, 2024b.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024c.

Yilun Liu, Yunpu Ma, Shuo Chen, Zifeng Ding, Bailan He, Zhen Han, and Volker Tresp. Perft: Parameter-efficient routed fine-tuning for mixture-of-expert model. *arXiv preprint arXiv:2411.08212*, 2024d.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Evan Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoe: Open mixture-of-experts language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xXTkbTBmqq.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In Gerardo Flores, George H Chen, Tom Pollard, Joyce C Ho, and Tristan Naumann (eds.), *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pp. 248–260. PMLR, 07–08 Apr 2022. URL https://proceedings.mlr.press/v174/pal22a.html.

Annemette Brok Pirchert, Jacob Nielsen, Mogens Henrik From, Lukas Galke Poech, and Peter Schneider-Kamp. Flexmore: A flexible mixture of rank-heterogeneous experts for efficient federatedly-trained large language models. *arXiv preprint arXiv:2602.08818*, 2026.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Yiru Tang, Kun Zhou, Xin Zhao, Jing Sha, Zhichao Sheng, and Shijin Wang. Exploring expert concentration for parameter-efficient fine-tuning of mixture-of-expert LLMs, 2026. URL https://openreview.net/forum?id=zBgjWTWgCh.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Qwen Team. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters", February 2024. URL https://qwenlm.github.io/blog/qwen-moe/.

Don Tuggener, Pius von Däniken, Thomas Peetz, and Mark Cieliebak. LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 1235–1241, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.155/.

An Wang, Xingwu Sun, Ruobing Xie, Shuaipeng Li, Jiaqi Zhu, Zhen Yang, Pinxue Zhao, Weidong Han, Zhanhui Kang, Di Wang, et al. Hmoe: Heterogeneous mixture of experts for language modeling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21954–21968, 2025.

Zihan Wang, Deli Chen, Damai Dai, Runxin Xu, Zhuoshu Li, and Yu Wu. Let the expert stick to his last: Expert-specialized fine-tuning for sparse architectural large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 784–801, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.46. URL https://aclanthology.org/2024.emnlp-main.46/.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 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, 2024. URL https://arxiv.org/abs/2309.12284.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Tong Zhu, Xiaoye Qu, Daize Dong, Jiacheng Ruan, Jingqi Tong, Conghui He, and Yu Cheng. Llama-moe: Building mixture-of-experts from llama with continual pre-training. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 15913–15923, 2024.

Statement on LLM Usage

All core research ideas, experimental design, data analysis, interpretation of results, and the final scientific conclusions presented in this paper were conceived and formulated by the human authors. The LLM's role was strictly that of an assistant for language and expression, not a contributor to the core scientific content or intellectual contributions of the work. The authors take full responsibility for all content in this manuscript, including any text that was assisted by an LLM.

Limitation

Although DR-LoRA demonstrates consistent improvements over strong baselines, several limitations remain. First, the saliency score is motivated by a first-order approximation and a mean-field factorization, and should therefore be viewed as a practical allocation heuristic rather than a strict optimal estimator of expert utility. Its behavior under more strongly coupled routing–gradient dynamics remains to be studied further. Second, our experiments are conducted primarily on text MoE LLMs with top-k routing. Whether the same design transfers equally well to other routing paradigms (e.g., expert-choice routing) or to multimodal MoE architectures is still an open question. In addition, dynamic rank growth requires pre-allocated rank space during training, which introduces extra training-time overhead. Understanding this trade-off more systematically at larger scales is an important direction for future work.

A Motivation for the Saliency Score

This appendix provides additional motivation for the saliency score design presented in §3.1 of the main paper, discusses the practical validity of the key approximation, and explains the design choices behind the rank importance measure.

A.1 From Capacity Expansion to the $f \cdot g$ Factorization

Under a fixed total training rank budget, the core allocation decision is: which expert should receive the next rank dimension to maximize the reduction in training loss? We address this question using a first-order Taylor approximation. Consider expanding the LoRA module of the expert $E_{\ell,i}$ by one rank dimension, introducing a parameter change $\Delta\boldsymbol{\theta}_{\ell,i}$. For an input token x, the first-order approximation of the loss change is:

$$\Delta\mathcal{L}(x) \;\approx\; \langle \nabla_{\boldsymbol{\theta}_{\ell,i}}\mathcal{L}(x),\; \Delta\boldsymbol{\theta}_{\ell,i}\rangle. \tag{7}$$

With a norm constraint on $\Delta\boldsymbol{\theta}_{\ell,i}$, the achievable first-order decrease depends on the alignment between the gradient direction and the parameter change, indicating that gradient magnitude is a natural proxy for the potential benefit of capacity expansion.

In dense models, every module processes all tokens and receives the correspondingly dense gradient signals. The key difference in MoE models is that expert i's output is weighted by the routing weight $z_{\ell,i}(x)$ before entering subsequent computation. The expert's parameter gradient is inherently modulated by its routing weight:

$$\nabla_{\boldsymbol{\theta}_{\ell,i}}\mathcal{L}(x) \;\propto\; z_{\ell,i}(x)\cdot \mathbf{h}_{\ell,i}(x), \tag{8}$$

where $\mathbf{h}_{\ell,i}(x)$ is the local gradient signal within the expert. When expert i is not selected by the top-k routing, $z_{\ell,i}(x) = 0$ and no gradient flows through it.

Treating $q_{\ell,i}(x) = \|\mathbf{h}_{\ell,i}(x)\|$ as a measure of the effective local gradient intensity when the expert is activated (consistent with the definition in §3.1), the expected loss reduction from expanding expert i's capacity can be written as:

$$\Delta\mathcal{L}_{\ell,i} \;\propto\; \mathbb{E}_x\big[z_{\ell,i}(x)\cdot q_{\ell,i}(x)\big]. \tag{9}$$

We decompose this expectation as follows:

$$\mathbb{E}[z_{\ell,i} \cdot q_{\ell,i}] \;=\; \mathbb{E}[z_{\ell,i}] \cdot \mathbb{E}[q_{\ell,i} \mid z_{\ell,i} > 0] \;+\; \mathrm{Cov}(z_{\ell,i},\, q_{\ell,i}). \tag{10}$$

Using a mean-field-style approximation that neglects the covariance term, we obtain:

$$\Delta\mathcal{L}_{\ell,i} \;\approx\; \underbrace{\mathbb{E}[z_{\ell,i}]}_{f_{\ell,i}\text{: routing frequency}} \cdot \underbrace{\mathbb{E}[q_{\ell,i} \mid z_{\ell,i} > 0]}_{g_{\ell,i}\text{: learning intensity when activated}}. \tag{11}$$

Validity of the approximation. The factorization in Eq. (11) is exact when $z_{\ell,i}$ and $q_{\ell,i}$ are independent; in practice, it requires their correlation to be weak relative to the product of their means. We emphasize that we do not claim that this holds exactly, rather, the $f \cdot g$ decomposition serves as a theoretically motivated heuristic that captures two distinct and complementary aspects of expert demand. Its practical value lies in whether the resulting ranking of experts leads to effective allocation decisions, not in whether the covariance term is strictly zero.

However, several aspects of the training setup reduce the magnitude of the covariance term. **(I)** Both f and g are computed as exponential moving averages over many training steps (Eqs. 2–4), which average out token-level fluctuations and substantially reduce the instantaneous correlation between routing weights and gradient intensity. **(II)** Rank growth decisions occur only every T_{grow} steps (default: 200), so the saliency scores that drive allocation are aggregated over hundreds of mini-batches, further smoothing any residual correlation. **(III)** In the default training settings, the router is frozen during warmup and unfrozen afterward (§B.2). When the router is trainable, the routing weights evolve and $\mathrm{Cov}(z, q)$ is, in principle, nonzero. However, in our training setup, the router is updated under the same optimizer schedule but represents a much smaller parameter than the expert LoRA modules. Therefore, the saliency statistics, which are further smoothed by EMA and updated only every T_{grow} steps, evolve more gradually than the token-level routing fluctuations. Moreover, the ablation in Table 3 shows that DR-LoRA with a frozen router (Avg 22.5) still outperforms both LoRA (Avg 21.1) and AdaLoRA (Avg 21.9) with an unfrozen router. This suggests that the saliency-based allocation remains useful even in the more stable routing regime induced by a frozen router. Unfreezing the router provides an additional benefit on top of the dynamic rank allocation, rather than being its sole source of improvement.

Semantics of the multiplicative. **(I)** Routing frequency $f_{\ell,i}$: how frequently the expert is used in task data. Higher frequency means that increasing this expert's capacity affects more training samples. **(II)** Conditional learning intensity $g_{\ell,i}$: how active the expert's gradients are when it is used. Even a frequently routed expert gains no benefit from additional capacity if its LoRA gradients are near zero (i.e., it has converged). A key property of the multiplicative form is that the score naturally approaches zero when either factor is near zero. This prevents two problematic allocation patterns: a frequently routed but converged expert (f large, $g \approx 0$) would not receive redundant capacity; a gradient-active but rarely routed expert ($f \approx 0$, g large) would not receive capacity based on noisy, sparse gradient signals. In contrast, an additive combination $f + g$ is more likely to assign high scores in both scenarios.

A.2 Rank Importance

The per-rank sensitivity $s^{(t)}_{\ell,i,j}$ in Eq. (3) and its expert-level aggregation $g^{(t)}_{\ell,i}$ in Eq. (4) involve several design choices, which we justify below.

Multiplicative form of per-rank sensitivity. The sensitivity measure $s_{\ell,i,j} = \|\nabla_{\mathbf{a}_j}\mathcal{L} \odot \mathbf{a}_j\|_1 \cdot \|\nabla_{\mathbf{b}_j}\mathcal{L} \odot \mathbf{b}_j\|_1$ adapts the gradient-weight importance criterion of Zhang et al. (2023) to LoRA's paired low-rank structure. In LoRA, the output contribution of the j-th rank dimension is $\mathbf{b}_j\,\mathbf{a}_j^\top\mathbf{x}$, which vanishes when either $\mathbf{a}_j$ or $\mathbf{b}_j$ is near zero. Accordingly, the multiplicative form ensures that a rank dimension is scored as important only when both

its A-side and B-side carry nontrivial gradient-weight products. An additive alternative $\|\nabla_{\mathbf{a}_j} \odot \mathbf{a}_j\|_1 + \|\nabla_{\mathbf{b}_j} \odot \mathbf{b}_j\|_1$ may assign nonzero importance even when one side has effectively converged, thereby overestimating the dimension's practical contribution.

Why average rather than sum or max. The expert-level aggregation $g_{\ell,i} = \frac{1}{r_{\ell,i}^{(t)}} \sum_j g_{\ell,i,j}$ uses the mean rather than the sum or maximum over active dimensions. This choice is driven by the allocation context: the saliency score determines which expert should receive the *next* rank dimension, so $g_{\ell,i}$ should reflect the per-dimension learning intensity (i.e., how much each existing dimension is still learning), not the total learning activity. Using the sum would scale linearly with $r_{\ell,i}^{(t)}$, biasing the score toward experts that already hold more ranks and creating a positive feedback loop. Using the maximum would make the score sensitive to outlier dimensions and vulnerable to gradient noise, particularly for low-frequency experts that accumulate only sparse gradient observations. The mean provides a natural rank-normalized statistic that allows fair comparison across experts with different current ranks.

Scale sensitivity and EMA smoothing. The multiplicative form of $s_{\ell,i,j}$ involves products of gradient-weight norms and is therefore sensitive to parameter scale. Two mechanisms mitigate this concern. First, the EMA smoothing in Eq. (4) averages out step-level fluctuations, producing stable estimates of the underlying learning intensity. Second, $g_{\ell,i}$ enters the saliency score only through the product $f_{\ell,i} \cdot g_{\ell,i}$, and the allocation operates via *relative ranking* within each layer (per-layer greedy in §3.3), not absolute thresholds. Therefore, global rescaling of sensitivities has a limited effect on the allocation outcome, since the allocation depends primarily on relative ranking within each layer rather than absolute thresholds.

A.3 Rank Penalty

The analysis in §A.1 considers the immediate value of adding one rank dimension, without accounting for how many ranks the expert already holds. Intuitively, the first few rank dimensions capture the dominant directions of the parameter update, while subsequent dimensions contribute progressively less, consistent with the rapid singular value decay commonly observed in low-rank matrix approximations. To model this effect, we assume that the incremental contribution of the r-th rank dimension decreases with r. Specifically, we model the total adaptation benefit of expert i as:

$$U_{\ell,i}(r) = \alpha_{\ell,i} \cdot \varphi(r), \qquad \alpha_{\ell,i} = f_{\ell,i} \cdot g_{\ell,i}, \tag{12}$$

where $\varphi(\cdot)$ is an increasing concave function ($\varphi' > 0$, $\varphi'' < 0$), indicating that each additional dimension provides a smaller incremental gain. In this model, the rank allocation problem becomes a discrete budget allocation problem: maximize $\sum_i \alpha_i \cdot \varphi(r_i)$ subject to $\sum_i r_i = \mathcal{B}$, where $\mathcal{B}$ is the total rank budget. For separable concave objectives, a greedy strategy that allocates each additional rank to the expert with the largest incremental gain is a natural and well-motivated choice. Choosing $\varphi'(r) = (r+1)^{-\gamma}$ with $\gamma > 0$, the incremental gain for expert i is:

$$\frac{\partial}{\partial r_{\ell,i}} U_{\ell,i}(r_{\ell,i}) \;=\; \frac{f_{\ell,i} \cdot g_{\ell,i}}{(r_{\ell,i}+1)^{\gamma}}, \tag{13}$$

which matches the saliency score $S_{\ell,i}$ used in Eq. (5). The parameter γ controls how strongly the growth priority decreases with the existing rank, which in turn determines the concentration of the final rank distribution: $\gamma = 0$ completely removes the penalty, while excessively large γ suppresses meaningful differentiation. We use $\gamma = 1.2$ as the default in all experiments.

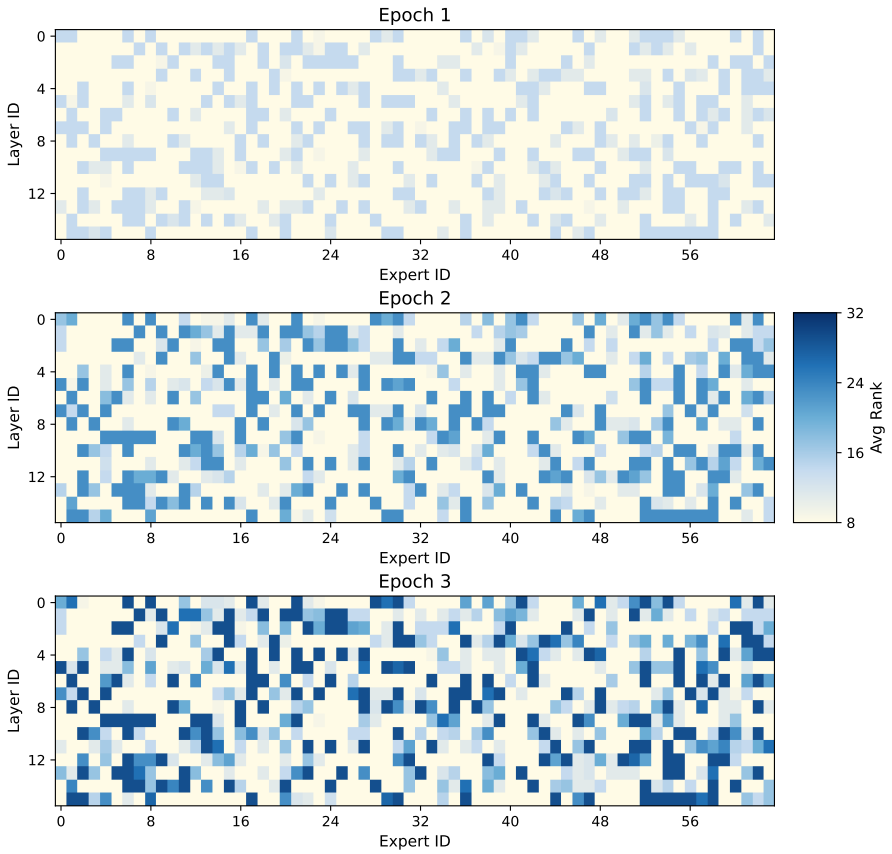


Figure 6: Evolution of expert LoRA ranks during DR-LoRA training on OLMoE. Each heatmap shows the average rank per expert at different training stages, with high-rank experts (darker blue) concentrated in task-relevant positions.

B Experimental Implementation

B.1 Training Datasets

We use six domain-specific datasets for fine-tuning:

- **MetaMathQA** (Yu et al., 2024) is a mathematical reasoning mixture dataset. We randomly sample 60k examples.
- **CodeAlpaca-20k** (Chaudhary, 2023) is a code instruction-following dataset containing approximately 20k samples. We use the full dataset for training.
- **OLMoE SFT Mix** (Muennighoff et al., 2025) is a general instruction-following mixture dataset used to train the OLMoE model. We randomly sample 60k examples.
- **MedMCQA** (Pal et al., 2022) is a large-scale medical multiple-choice QA dataset. We randomly sample 60k examples from the train split.
- **WMT-DA-Human-Evaluation** is a machine translation dataset containing DA human annotations from WMT News Translation shared tasks (2017–2022) (Bojar et al., 2017; 2018; Barrault et al., 2019; 2020; Akhbardeh et al., 2021; Kocmi et al.,

https://huggingface.co/datasets/meta-math/MetaMathQA
https://huggingface.co/datasets/sahil2801/CodeAlpaca-20k
https://huggingface.co/datasets/allenai/tulu-v3.1-mix-preview-4096-OLMoE
https://github.com/MedMCQA/MedMCQA
https://huggingface.co/datasets/RicardoRei/wmt-da-human-evaluation

Hyperparameter	Value
Initial rank (r_{init})	8
Final Avg rank (r_{target})	16
Maximum rank (r_{max})	32
Growth interval (T_{grow})	200
Module growth fraction (p_{grow})	0.1
Usage EMA coefficient (β)	0.9
Rank penalty exponent (γ)	1.2
Learning rate	2×10^{-5}
LR scheduler	Linear
Warmup ratio	0.03
Weight decay	0.0
Optimizer	AdamW (fused)
LoRA α	$2 \times r$
Epochs	3
Total steps	3,750
Micro-batch size	3
Gradient accumulation	4
Effective batch size	48
Max sequence length	512 - 4096 (depending on the training datasets)

Table 7: Complete hyperparameter settings for all experiments.

2022). We sample 60k examples across three language pairs (en-cs, en-de, en-zh, 20k each).

- **LEDGAR** (Tuggener et al., 2020) is a multilabel corpus of legal provisions in contracts. We reformat it as a multiple-choice QA task and train on the full 60k examples.

B.2 Training Configurations

B.2.1 Training Hyperparameters

Table 7 presents the complete hyperparameter settings for all experiments.

B.2.2 Growth Schedule and Router Training

Growth window (T_{window}): Rank growth begins after the learning rate warmup phase (t_{warmup}) and stops 200 steps before training completion (t_{end}), and a growth window can therefore be calculated as $T_{\text{window}} = t_{\text{end}} - t_{\text{warmup}}$. This ensures that ranks activated at the final growth step still receive at least 200 steps of training before the process concludes, allowing newly added capacity to be sufficiently optimized.

Layer synchronization: All layers grow simultaneously at each growth event. The per-layer quota is computed as $Q = \lceil N \times (r_{\text{target}} - r_{\text{init}})/N_{\text{grow}} \rceil$. Taking OLMoE as an example, N=128 is the number of LoRA modules per layer (64 experts $\times$ 2 projections) and N_{grow} is the number of scheduled growth events, determined by T_{window} and T_{grow} as $N_{\text{grow}} = \lfloor T_{\text{window}}/T_{\text{grow}} \rfloor$.

Router training schedule: The MoE router remains frozen during the warmup phase to stabilize LoRA training. After warmup, the router is unfrozen and trained jointly with LoRA modules until training completion. This allows the router to adapt to the evolved expert capabilities from dynamic rank allocation.

https://huggingface.co/datasets/coastalchp/ledgar

Domain	Benchmark	Metric	Few-shot	Split	Size
Math	GSM8K	Accuracy	8-shot	test	1,319
Code	HumanEval	Pass@1	0-shot	test	164
IF	IFEval	Accuracy	0-shot	test	541
Med QA	MedMCQA	Accuracy	0-shot	test	6,150
Machine Translation	WMT23	COMET	0-shot	test	4,697
Legal Understanding	LEDGAR	Accuracy	0-shot	test	10,000

Table 8: Evaluation settings for all benchmarks.

B.2.3 Computational Infrastructure

All experiments are conducted on a single server with 4× NVIDIA L40S GPUs (48GB each). We use DeepSpeed ZeRO-2 for distributed training with bfloat16 mixed precision, and Flash Attention 2 for memory-efficient attention computation.

B.3 Evaluation Settings

All evaluations are conducted using vLLM for efficient inference, except for LLaMA-MoE, which is not supported by vLLM and is therefore evaluated using the HuggingFace Transformers backend. All evaluations are conducted with 3 independent runs using random seeds $\{42, 123, 456\}$ and sampling temperature $\tau = 0.2$. Table 8 summarizes the evaluation configuration for each benchmark.

C Additional Experimental Results

C.1 Statistical Significance of the Main Results

To further assess the consistency of the improvements reported in Table 1, we conduct statistical tests at both the individual-setting and aggregate levels. For each of the 18 model–task settings, we compare DR-LoRA with the best-performing competing baseline in that setting using a paired t-test over three matched random seeds. Table 9 reports the corresponding mean scores, performance differences, and p-values. We additionally perform an aggregate paired t-test by treating the DR-LoRA score and its strongest-baseline counterpart in each model–task setting as one paired observation.

As shown in Table 9, DR-LoRA outperforms the best-performing competing baseline in all 18 model–task settings, with performance gains ranging from 1.0 to 2.5 points. The mean improvement across settings is 1.8 points, and the aggregate paired test confirms that the improvement is statistically significant ($p < 0.001$). At the individual-setting level, 12 of the 18 comparisons are significant at $p < 0.05$. The remaining comparisons are primarily associated with WMT23 and MedMCQA, where the differences among the strongest methods are relatively small. Nevertheless, all setting-level differences remain positive, indicating that the aggregate improvement is not driven by a small subset of models or downstream tasks.

C.2 Initial Rank Ratio Analysis

The initial active rank r_{init} determines each expert's starting capacity before dynamic growth begins. A smaller r_{init} allows more room for saliency-guided differentiation but risks under-training early; a larger r_{init} ensures stable early training but limits the growth budget for reallocation. We evaluate three settings on OLMoE with fixed r_{target}=16 and $r_{\max}$=32. All three configurations outperform LoRA, confirming robustness to this setting. The default r_{init}=$r_{\text{target}}/2$ delivers the best performance: it reserves half the budget for dynamic reallocation while providing enough initial capacity for stable early-stage learning. Too small ($r_{\text{target}}/4$): experts start with minimal capacity, causing slower convergence early before growth events begin. Too large ($3r_{\text{target}}/4$): the limited growth budget (only 4 ranks per expert) restricts achievable heterogeneity, reducing the benefit of dynamic allocation.

Model	Task	Best Baseline	DR-LoRA	Diff.	p-value
OLMoE	GSM8K	AdaLoRA (26.3)	28.4	+2.1	0.039*
	HumanEval	PERFT-R (15.5)	16.7	+1.2	0.068
	IFEval	PERFT-R (24.3)	26.7	+2.4	0.022*
	MedMCQA	AdaLoRA (41.5)	43.9	+2.4	0.035*
	WMT23	PERFT-R (64.0)	66.5	+2.5	0.213
	LEDGAR	AdaLoRA (80.1)	82.1	+2.0	0.009**
Qwen1.5-MoE	GSM8K	PERFT-R (65.6)	67.2	+1.6	0.028*
	HumanEval	AdaLoRA (45.2)	46.5	+1.3	0.041*
	IFEval	PERFT-R (33.3)	35.1	+1.8	0.007**
	MedMCQA	PERFT-R (48.3)	50.1	+1.8	0.259
	WMT23	AdaLoRA (79.3)	80.7	+1.4	0.069
	LEDGAR	PERFT-R (93.4)	94.8	+1.4	0.016*
LLaMA-MoE	GSM8K	AdaLoRA (13.6)	15.7	+2.1	0.008**
	HumanEval	PERFT-R (11.1)	13.1	+2.0	0.013*
	IFEval	AdaLoRA (21.2)	23.4	+2.2	0.012*
	MedMCQA	PERFT-R (31.3)	32.9	+1.6	0.059
	WMT23	AdaLoRA (64.8)	65.8	+1.0	0.098
	LEDGAR	PERFT-R (80.0)	81.3	+1.3	0.019*
Aggregate across 18 settings			–	**+1.8**	$<$ **0.001**

Table 9: Statistical comparison between DR-LoRA and the best-performing competing baseline in each model–task setting. Reported method scores are averages over three random seeds. Per-setting p-values are obtained using paired t-tests over the three matched seeds. The aggregate test treats the 18 model–task settings as paired observations. $^{*}p < 0.05$ and $^{**}p < 0.01$.

r_{init}	Growth Budget	**GSM8K**	**HumanEval**	**IFEval**	**Avg**
4 ($r_{\text{target}}/4$)	12 per expert	26.8	15.6	24.9	22.4
8 ($r_{\text{target}}/2$)	**8 per expert**	**28.4**	**16.7**	**26.7**	**23.9**
12 ($3r_{\text{target}}/4$)	4 per expert	27.3	15.9	25.6	22.9
LoRA (r=32, fixed)	0	25.2	14.8	23.3	21.1

Table 10: Impact of initial rank r_{init} on OLMoE. All settings outperform LoRA (21.1 avg). r_{init}=$r_{\text{target}}/2$ provides the best trade-off.

C.3 Expert Rank Evolution Visualization

To illustrate how DR-LoRA dynamically constructs heterogeneous rank distributions, we visualize the evolution of expert LoRA ranks throughout training in Figure 6. At the early stage (Epoch 1), most experts remain at the initial rank $r_{\text{init}} = 8$ (shown in light cream), with only a few high-saliency experts receiving additional capacity (darker colors). By mid-training (Epoch 2), a clear heterogeneous pattern emerges as DR-LoRA progressively allocates ranks to task-relevant experts based on routing frequency and rank importance. At the final stage (Epoch 3), the rank distribution becomes highly differentiated, with some experts reaching the maximum rank $r_{\text{max}} = 32$ (deep navy blue) while others remain at lower ranks, reflecting their varying importance to the target task. This progression demonstrates DR-LoRA 's ability to automatically discover and amplify task-relevant experts through dynamic capacity allocation, forming a task-adaptive structure without manual intervention.

D Computational Cost Analysis

D.1 Memory Analysis

We empirically measure the GPU memory footprint of DR-LoRA under our experimental configuration (OLMoE, 4×L40S GPUs, DeepSpeed ZeRO-2, bfloat16 mixed precision). As

Method	Memory / GPU	Overhead vs. LoRA (r=16)
LoRA (r=16)	∼38 GB	—
LoRA (r=32)	∼40 GB	+2 GB
DR-LoRA (r_{init}=8, r_{target}=16)	∼40 GB	+2 GB

Table 11: Per-GPU memory usage under ZeRO-2 training on OLMoE with 4×L40S GPUs. DR-LoRA matches LoRA (r=32) exactly, as both allocate identical parameter space.

Task	Method	Time	Overhead	Acc	ΔAcc
GSM8K	Base	-	-	12.9	-
	LoRA ($r = 32$)	9.6h	-	25.2	+12.3
	DR-LoRA ($r_{init} = 8, r_{target} = 16$)	**10.1h**	**5%**	**28.4**	**+15.5**
HumanEval	Base	-	-	13.6	-
	LoRA ($r = 32$)	5.5h	-	14.8	+1.2
	DR-LoRA ($r_{init} = 8, r_{target} = 16$)	**5.7h**	**4%**	**16.7**	**+3.1**

Table 12: Wall-clock training time comparison across two tasks. Overhead is relative to LoRA ($r = 32$). ΔAcc denotes accuracy gain over the base model.

shown in Table 11, DR-LoRA incurs **identical GPU memory usage to LoRA (r=32)**, and introduces approximately 2 GB overhead per GPU compared to LoRA (r=16).

This equivalence is expected: DR-LoRA pre-allocates LoRA parameter space up to $r_{\max}$=32 for all experts, which is identical to the parameter count of LoRA (r=32). Under ZeRO-2, parameters and gradients (bf16) are replicated across devices while optimizer states (fp32) are sharded. The 2 GB overhead relative to LoRA (r=16) thus reflects the additional 100.6M parameters allocated for the reserved rank dimensions ($r_{\max} - r_{\text{target}} = 16$ dimensions per expert). Crucially, this memory footprint is identical to the primary baseline we compare against (LoRA r=32), meaning DR-LoRA achieves its performance gains without requiring any additional memory resources beyond LoRA at the matched parameter budget.

D.2 Training Time Analysis

We measure wall-clock training time using 4×L40S GPUs across two training scenarios: mathematical reasoning, where we train OLMoE for three complete epochs on the MetaMathQA dataset and evaluate on GSM8K, and code generation, where we train on CodeAlpaca-20K and evaluate on HumanEval. Results are summarized in Table 12. DR-LoRA incurs a modest training time overhead of 5% and 4% over LoRA ($r = 32$) on MetaMathQA and CodeAlpaca-20K respectively. Yet despite this near-equivalent computational cost, DR-LoRA achieves substantially larger performance gains: +15.5 points on GSM8K versus +12.3 for LoRA ($r = 32$), and +3.1 points on HumanEval versus +1.2 respectively. This consistent pattern across both mathematical reasoning and code generation demonstrates that dynamic rank allocation yields substantially greater performance gains than simply scaling up a static rank. The overhead is attributable to dynamic rank allocation mechanisms, including importance scoring, expert usage tracking, and periodic rank growth.

D.3 FLOPs Analysis

We analyze the computational cost of DR-LoRA in terms of floating-point operations (FLOPs) during training. The total FLOPs consist of base expert computation and LoRA adaptation. For a single forward pass, base expert FLOPs are:

$$\text{FLOPs}_{\text{base}} = 4BLK \cdot d_m \cdot d_e \tag{14}$$

where B=4096 is the effective batch size, L=16 is the number of layers, K=8 is the number of activated experts per layer, d_m=2048 is the hidden dimension, and d_e=1024 is the expert

Method	Base	LoRA	Total	Rel.
No LoRA	4398.0	—	4398.0	1.000×
LoRA (r=32)	4398.0	137.4	4535.5	1.031×
DR-LoRA (r_{init}=8, r_{target}=16)	4398.0	137.4	4535.5	1.031×

Table 13: FLOPs per sample (GFLOPs) for forward pass. Base expert computation dominates (97–99%), making LoRA's contribution minimal. DR-LoRA's LoRA FLOPs are computed using the maximum rank (r=32) as a conservative upper bound.

dimension. LoRA adds:

$$\text{FLOPs}_{\text{LoRA}} = 8BLK \cdot d_e \cdot r \tag{15}$$

where r is the LoRA rank and each expert has two LoRA modules (`up_proj` and `down_proj`). Table 13 presents the FLOPs analysis for different methods. DR-LoRA has identical LoRA FLOPs to LoRA (r=32), as we use the maximum rank for a conservative upper-bound estimate. Since the base expert computation dominates total FLOPs (98.5%), the overall increase over the no-LoRA baseline is only 3.1%.