ID-LoRA: Efficient Low-Rank Adaptation Inspired by Matrix Interpolative Decomposition

Anonymous EMNLP2026 submission

Abstract

LoRA has become a universal Parameter-Efficient Fine-Tuning (PEFT) technique that equips Large Language Models (LLMs) to adapt quickly to new tasks. However, when these models are scaled up, even the latest LoRA variants still introduce considerable overhead in trainable parameters. Conversely, aggressively lowering the rank to curb this overhead markedly degrades performance in complex multi-task settings. We propose ID-LoRA, a novel PEFT framework that breaks the trade-off. Its core innovation lies in extracting and reusing clustered parameter groups from the pretrained weight matrix. These groups are then used to form multiple low-rank components, all of which share only a single initialized trainable low-rank matrix. This approach cuts the number of trainable parameters while keeping the model's capacity intact. We evaluate ID-LoRA on five diverse benchmarks: Mathematical Reasoning, Code Generation, MMLU, CommonsenseQA, and Safety Alignment. ID-LoRA outperforms both full fine-tuning and existing PEFT baselines (e.g., LoRA, DoRA, HydraLoRA) while using up to 46% fewer trainable parameters than the standard LoRA. In multi-task scenarios, it surpasses LoRA and its recent variants (e.g., DoRA and HydraLoRA) on both Code and MMLU tasks, yet requires only 54% of the trainable parameters demanded by the conventional LoRA.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Dubey et al., 2024; Yang et al., 2024) exhibit strong cross-task transfer capabilities, driving widespread adoption of a unified multi-task adaptation framework (Fang et al., 2024; Ding et al., 2023). In these frameworks, a single foundational LLM is fine-tuned to meet diverse downstream requirements. However, full fine-tuning is costly. Low-Rank Adaptation (LoRA) (Hu et al., 2022) has become the standard choice because it is efficient. Yet it faces a critical trade-off: higher ranks linearly increase the number of trainable parameters and memory use, while a fixed low-rank setting fails to adapt to multi-task scenarios.

To address this limitation, we propose ID-LoRA, a novel low-rank PEFT framework that fundamentally rethinks parameter efficiency. Inspired by Matrix Interpolative Decomposition (MID) (Ho and Greengard, 2012), ID-LoRA first performs Parameter Matrix Row Clustering (PMRC): it clusters rows of the frozen pretrained weight matrix $W \in \mathbb{R}^{d\times d}$ (See Figure 1 (b)) into k task-specific bases $\{A_i\}_{i=1}^{k}$ using k-means constrained with minimum cluster size (Levy-Kramer, 2018) under Euclidean distance. These low-rank matrices $\{A_i\}_{i=1}^{k}$ are then frozen, and a shared trainable matrix B is learned to jointly compose the PEFT update. To further improve multi-task modeling under strict parameter budgets, we introduce Rank Boosting (RB), which increases the effective rank of the update via feature splitting and concatenation while avoiding an increase in trainable parameters. Concretely, RB applies a shared projection and shrinks the trainable matrix B from $\mathbb{R}^{d\times r}$ to $\mathbb{R}^{\frac{d}{s}\times\frac{r}{s}}$.

The key innovation of ID-LoRA is to reuse clustered parameters from pretrained weights as frozen bases and train only a shared matrix B to compose them (Figure 1). This enables higher effective ranks (e.g., r=128) with fewer trainable parameters. ID-LoRA's multi-task gains are supported by theory: Theorem 2 shows that when m tasks exhibit a k-cluster structure (Assumption 1), the clustering-derived bases $\{A_i\}_{i=1}^{k}$ achieve tighter error bounds than global low-rank adaptation (LoRA).

Experiments validate both effectiveness and efficiency. In single-task settings, ID-LoRA surpasses all baselines on mathematics, code generation, and safety tasks while updating only 0.56% of the parameters of LLaMA-3-8B. In multi-task instruction

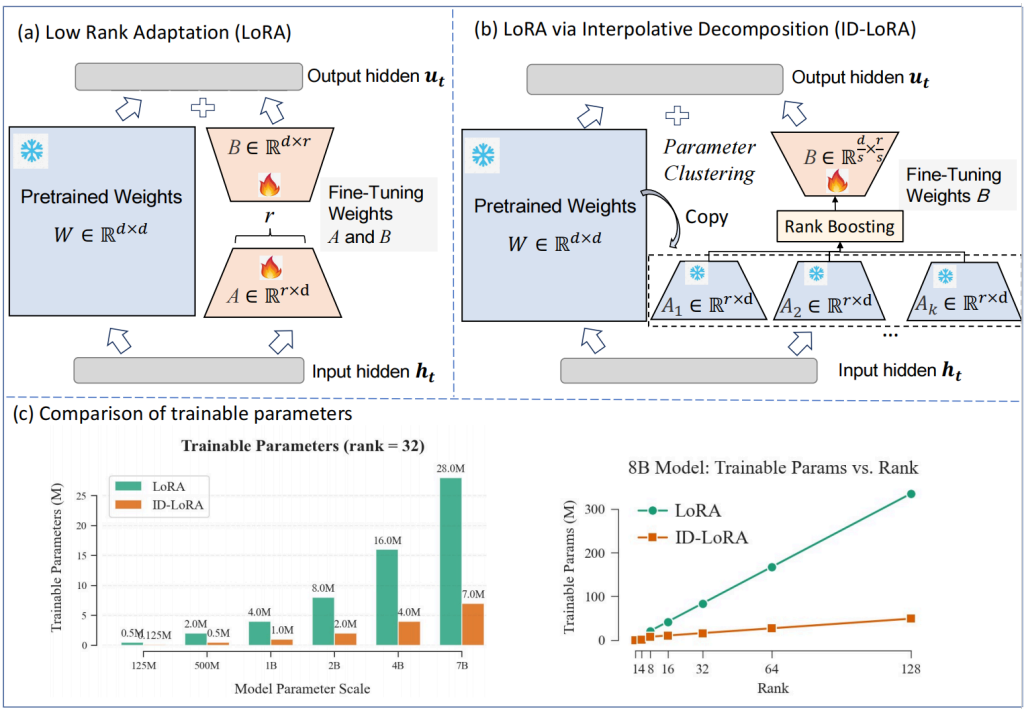


Figure 1: Architectural Comparison and Parameter Efficiency: LoRA versus ID-LoRA. (a) LoRA requires training two low-rank matrices: randomly initialized $A \in \mathbb{R}^{r\times d}$ and zero-initialized $B \in \mathbb{R}^{d\times r}$. (b) ID-LoRA employs the parameter clustering and rank boosting to generate multiple low-rank components while sharing a single B, thereby reducing trainable parameters. (c) Trainable parameters: ID-LoRA achieves $\sim 5\times$ compression versus LoRA at rank 32 (right) and maintains superior scalability across model sizes (left).

tuning on MMLU, ID-LoRA reduces trainable parameters by 46% versus LoRA and yields a 6.0% absolute average gain on LLaMA-3-8B. We further evaluate on Math, CommonsenseQA, and Code Generation, where ID-LoRA outperforms baselines on two of the three benchmarks. The main contributions of our work are as follows: (1) ID-LoRA reuses frozen weights as low-rank bases and trains only a single shared matrix, eliminating the need for matrix A; RB further boosts effective rank for multi-task modeling without increasing trainable parameters. (2) We prove that clustering decomposition yields tighter error bounds and more stable pivots, ensuring greater reliability in multi-task scenarios. (3) ID-LoRA achieves or surpasses LoRA-level accuracy while reducing trainable parameters by up to 46%.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT). PEFT aims to adapt LLMs to downstream tasks with minimal parameter changes. This line of research has led to a wide range of advancements (Houlsby et al., 2019; Hu et al., 2022; Huang et al., 2023; Tian et al., 2024). The three typical PEFT methods include **Adapter** (Houlsby et al., 2019), **Prefix-tuning** (Li and Liang, 2021) and **LoRA** (Hu et al., 2022). Adapter achieves adaptation to new tasks by inserting small adapter layers between the pretrained layers, without retraining the entire model. This method has been used in various domains (Pfeiffer et al., 2021; Qiao et al., 2023). Prefix-tuning optimizes task-specific prefix vectors.

LoRA popularized low-rank decomposition for weight updates, balancing trainable parameters and performance. However, LoRA faces a critical trade-off: higher ranks improve task adaptation but linearly increase parameters and memory, while lower ranks degrade performance (Zhang et al., 2023, 2025). Several studies have proposed LoRA variants (Liu et al., 2024; Zhang et al., 2025; Kopiczko et al., 2024) to enhance learning capacity and reduce the number of trainable parameters. Other studies (He et al., 2022) have provided a unified perspective, viewing many PEFT methods as a form of adapter. However, these techniques show constraints in multi-task scenarios.

Multi-LoRA Adapters. Researchers have explored the benefits of modeling multiple LoRA adapters for multi-task. LoRAHub (Huang et al., 2023) adopts a multi-adapter approach by training several adapters and selecting domain-specific com-

binations at inference. MoELoRA (Liu et al., 2023) leverages the Mixture-of-Experts (MoE) (Jacobs et al., 1991) to model multi-domain knowledge. HydraLoRA (Tian et al., 2024) uses an asymmetric structure to manage multiple LoRA adapters and enhances parameter efficiency compared to symmetric approaches. These methods, however, necessitate training multiple adapters and consequently introduce additional trainable parameters. Different from previous methods, the core innovation of ID-LoRA lies in constructing a frozen matrix by clustering the pretrained parameters, rather than initializing multiple independent LoRA adapters.

3 Preliminaries

3.1 Low-Rank Adaptation

Low-Rank Adaptation (LoRA) (Hu et al., 2022) introduces the concept of "intrinsic rank" in the parameter update process. This rank represents the true degrees of freedom needed for task adaptation. It is far smaller than the total parameter count, yet sufficient for effective fine-tuning. The insight is that pretrained language models have a minimal intrinsic dimension. Fine-tuning within this small subspace matches the effect of adjusting the entire parameter space (Aghajanyan et al., 2021). For the weight matrix W, its parameter update ΔW can be represented by low-rank decomposition A and B. In the fine-tuning phase, the adjustment is confined to low-rank matrices A and B, while all other model parameters remain immutable. The update mechanism for these matrices is delineated in the subsequent equation:

$$u_t = Wh_t + \Delta Wx = Wh_t + \frac{\alpha}{r}BAh_t \tag{1}$$

where h_t denotes the hidden state at token position t, α is the scaling factor, $W \in \mathbb{R}^{d\times d}$, $A \in \mathbb{R}^{r\times d}$ and $B \in \mathbb{R}^{d\times r}$, and $r < d$. To align with LoRA's initialization and ensure that ΔW is zero at the start of training, matrix B in ID-LoRA is initialized to all zeros.

3.2 Matrix Interpolative Decomposition

Matrix Interpolative Decomposition (MID) (Horn and Johnson, 2012) is a structured low-rank matrix decomposition method: it identifies key rows (or columns) as a skeleton and approximates the original matrix via multiplication with a small low-rank matrix. In our work, we approximate ΔW by selectively reusing pretrained parameter matrix entries, reducing trainable parameters. For MID, we can write $\Delta W = BA$, where $A = W_{[S,:]}$ is formed by taking the rows of W indexed by the set S, and B is the low-rank coefficient matrix to be learn. In this way, a small number of rows of W serve as the $skeleton$ that approximates the matrix ΔW. Specifically, the optimal row subset S and the matrix B are obtained by solving the following problem:

$$arg\,min||BW_{[S,:]} - \Delta W||_F \tag{2}$$

where $|S| = r$, and $\|\cdot\|_F$ is the matrix 2-norm. Selecting key rows $W_{[S,:]}$ in MID is non-trivial. We cluster rows into subsets that serve as low-rank skeletons for weighted ΔW approximation.

4 Method

Figure 2 summarizes the overall architecture and how PMRC, routing, and RB interact during the forward pass.

4.1 Parameter Matrix Row Clustering

Given a parameter matrix $W \in \mathbb{R}^{d\times d}$, the rows of W are treated as individual elements within a collection. These row vectors are partitioned into k distinct clusters using the K-Means constrained with minimum cluster size (Levy-Kramer, 2018) algorithm with the Euclidean distance metric. Within each cluster $C_i (i = 1, 2, ..., k)$, we select the r row vectors exhibiting the smallest Euclidean distance to the cluster centroid μ_i. The selected rows from each cluster C_i are then aggregated to form a low-rank matrix $A_i \in \mathbb{R}^{r\times d}$. This process yields a set of k structured low-rank matrices $A_i, A_2, \ldots, A_k$ that collectively capture the dominant row-space patterns of the original matrix W, prioritizing proximity to cluster centers. In our experiments, k is equal to 4, and r is set according to the specific experimental requirements. We provide a comparative analysis justifying MID initialization against SVD and random initialization in Appendix B.4.

To apply these frozen bases during adaptation, we introduce an initialized trainable shared matrix $B \in \mathbb{R}^{d\times r}$ and use a lightweight router to dynamically compose the update across experts.

$$u_t = Wh_t + \sum_{i=1}^{k} \alpha_i B(A_i h_t) \tag{3}$$

where h_t denotes the input hidden state at token position t, and α_i is the router weight for expert i. Specifically, let $x_i = A_i h_t \in \mathbb{R}^r$ and let $T \in \mathbb{R}^r$ be a trainable routing vector; we compute

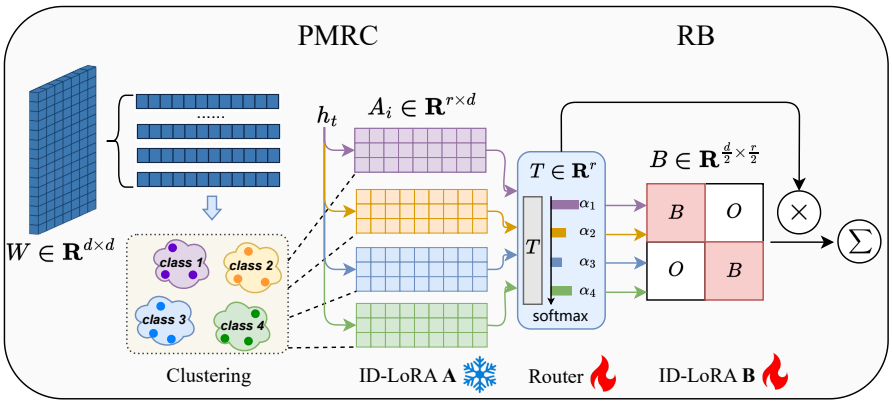


Figure 2: Overview of ID-LoRA. PMRC clusters rows of the pretrained weight matrix to form k frozen bases $\{A_i\}_{i=1}^k$. Given the hidden state h_t at token position t, ID-LoRA computes expert features $x_i = A_i h_t$ and uses a lightweight router with a trainable vector T to produce mixture weights $\{\alpha_i\}_{i=1}^k$. Rank Boosting applies a shared matrix B to split features and concatenates the outputs to form the final update.

$\alpha_i = \mathrm{softmax}_i(\langle T, x_i\rangle)$, where $\langle\cdot,\cdot\rangle$ denotes the inner product. This design reuses clustered row patterns while allowing input-adaptive composition of multiple frozen bases. Appendix B.9 reports router activation statistics across task types, revealing systematic differences that motivate learnable routing weights rather than fixed α values.

4.2 Rank Boosting

Relying on a single shared weight matrix B significantly complicates achieving balanced and harmonious interactions among expert modules. RB is designed to increase the effective rank of the composed update for multi-task modeling while keeping the trainable-parameter count fixed by reusing a single shared projection. To address this issue, we first partition the activation vector $x \in \mathbb{R}^r$ through matrix multiplication with A_i, then integrate these partitioned activation vectors via this unified shared low-rank matrix B, as shown in Figure 2. The formula is as follows:

$$x_i = A_i h_t,\ u_t = W h_t + \sum_{i=1}^{k} \alpha_i f_c([Bx_i^1, Bx_i^2]) \quad (4)$$

where $B \in \mathbb{R}^{\frac{d}{2}\times\frac{r}{2}}$, $A_i \in \mathbb{R}^{r\times d}$, and f_c is the concatenation operator. x_i^1 and x_i^2 are equal divisions from vector x_i. α_i is the router output.

Rank Analysis. Compared with LoRA and its multi-task-oriented variant MoELoRA (Liu et al., 2023), our method simultaneously reduces the number of trainable parameters and preserves a higher rank. MoELoRA instantiates k independent adapters $A_i \in \mathbb{R}^{r\times d}$ and $B_i \in \mathbb{R}^{d\times r}$, giving $k \times r \times (d+d)$ trainable parameters and, by the sub-additivity of matrix rank,

$$Rank(A_1 + A_2) \leq Rank(A_1) + Rank(A_2) \quad (5)$$

For a single LoRA adapter, the number of trainable parameters is $r \times (d+d)$, and its rank is r. In contrast, we introduce a single low-rank matrix $B \in \mathbb{R}^{\frac{d}{2}\times\frac{r}{2}}$ and a lightweight router $T \in \mathbb{R}^r$. The parameter count is therefore only $(r \times d)/4 + r$. During the forward pass, the outputs of k fixed matrices $\{A_1, A_2, \ldots, A_k\}$ are multiplied with B and then concatenated, yielding a representation whose rank is at most $k \times r$. Consequently, RB enables ID-LoRA to increase effective rank via concatenation without increasing trainable parameters, and attains a rank superior to that of both LoRA and MoELoRA.

4.3 Theoretical Analysis of Clustered Decomposition

The stability and quality of low-rank approximations depend heavily on the selection of "pivots" (representative rows or columns). In this section, we analyze the Parameter Matrix Row Clustering (PMRC) algorithm through the lens of Matrix Interpolative Decomposition (specifically CUR decomposition). We establish a theoretical connection where the rows selected by our clustering algorithm correspond to the "pivots" in the CUR framework. Although our approach—using k-means clustering to identify task-specific pivots—differs from traditional statistical selection methods (e.g., leverage scores), we prove it effectively addresses the same core challenge: ensuring high-quality and stable pivot selection in multi-task settings. The following analysis formally demonstrates two principal strengths of this approach: (1) enhanced multi-task learning via multiple low-rank subspaces, and (2) robustness to pivot selection.

First, we introduce three preliminary assumptions: (1) Task Clustering Structure: the optimal parameters of related tasks form disjoint groups; (2) Task Parameter Sharing: a single base matrix can be composed on-the-fly to yield task-specific parameters; (3) Pivot Sensitivity: the accuracy of standard low-rank factorizations, specifically the Column-Row-Update (CUR) decomposition as formalized by Penrose and Todd (1956), demonstrates dependence on the optimal selection of pivot indices during basis construction.

Assumption 1 (Task Clustering Structure) *Let $\{W_i^*\}_{i=1}^m \subset \mathbb{R}^{d_1 \times d_2}$ denote the optimal parameter matrices for m related tasks. We assume these matrices can be partitioned into k disjoint clusters $\mathcal{C}_1, \mathcal{C}_2 \ldots, \mathcal{C}_k$ (with $k < m$) such that:*

$$\forall i, j \in \mathcal{C}_l, \quad \textit{Rank}(W_i^* - W_j^*) \leq r_l < \min(d_1, d_2) \quad (6)$$

where $\mathcal{C}_l$ denotes the l-th cluster and r_l is the intrinsic rank difference within the cluster.

Assumption 2 (Task Parameter Sharing) *We assume the existence of a shared feature matrix $B \in \mathbb{R}^{d_2 \times r_g}$ with $r_g < d_2$, such that every optimal task-specific parameter matrix can be written as:*

$$W_i^* = BA_i + E_i, \quad \|E_i\|_F^2 \leq \epsilon \quad (7)$$

where $A_i \in \mathbb{R}^{r_g \times d_1}$ is a task-specific and E_i is the reconstruction error.

Definition 1 (Pivot Sensitivity) *The* CUR *decomposition's approximation accuracy crucially depends on pivot selection, with reconstruction errors escalating sharply when the pivots inadequately capture the principal row or column subspaces,*

$$\max_{\textit{bad pivots}} \|W - \tilde{W}_{CUR}\|_F^2 > \min_{\textit{good pivots}} \|W - \tilde{W}_{CUR}\|_F^2 \quad (8)$$

Then, we proceed to formalize the guarantees of the PMRC through two theorems, Clustering Reconstruction and Cluster-Pivot Stability.

Theorem 1 (Clustering Reconstruction) *Under Assumptions 1 and 2, the clustering-aware decomposition achieves a tighter reconstruction error bound compared to global low-rank decomposition:*

$$\sum_{i=1}^{m} \|W_i^* - \tilde{W}_i\|_F^2 \leq \sum_{i=1}^{m} \|W_i^* - \tilde{W}_i^{global}\|_F^2 - \Delta \quad (9)$$

where $\Delta = \sum_{l=1}^{k} \sum_{i \in \mathcal{C}_l} \|B(A_{l(i)} - A^{global})\|_F^2 \geq 0$. The inequality becomes strict ($\Delta > 0$) when tasks exhibit clustering structure.

Theorem 2 (Cluster-Pivot Stability) *Under Definition 1, the clustering-based local pivot selection ensures a tighter expected error bound compared to global pivot selection:*

$$\mathbb{E}_{\mathcal{P}} \left[\max_{1 \leq i \leq m} \|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F^2 \right] \geq \mathbb{E}_{\mathcal{P}_l} \left[\max_{1 \leq i \leq m} \|W_i^* - \tilde{W}_i\|_F^2 \right], \quad (10)$$

where $\tilde{W}_i^{\text{CUR}}$ is the reconstruction parameter using global pivots on all tasks, $\tilde{W}_i$ is reconstruction parameter using cluster-local pivots (per-cluster CUR *on $\mathcal{C}_l$), and $\mathcal{P}_l$ is the set of pivots selected from cluster $\mathcal{C}_l$'s centroid matrix.*

Our theory links cluster local pivot stability to ID LoRA effectiveness. Theorem 1 shows cluster aware decomposition beats global approximation when tasks cluster. Theorem 2 shows local pivot selection within clusters lowers worst case error sensitivity versus global selection. Our method picks rows closest to centroids and Theorem 2 confirms this gives theoretical stability. These results prove our clustering approach gives tight error bounds and robust multi task performance. Proofs are in Appendix A.

5 Experiments

To comprehensively evaluate ID-LoRA, we conduct experiments as follows: (1) Single-task experiments assessing general adaptation capability. (2) Multi-task experiments testing cross-domain transferability. (3) Efficiency analysis of time and memory. (4) Ablation studies on core components (i.e., PMRC and RB) and a performance comparison between LoRA and ID-LoRA under comparable trainable-parameter budgets. We conducted additional experiments to evaluate how the number of pretrained parameter clusters affects ID-LoRA's performance in multi-task scenarios. These results, along with other findings not presented in the main paper, are provided in Appendix B.3.

5.1 Experimental Setup

We conduct systematic comparisons across both single-task and multi-task settings to holistically evaluate ID-LoRA's adaptability and scalability.

Single-Task. We evaluate three capabilities: Math, Code, and Safety. (i) Mathematical Reasoning (**Math**): We fine-tune on the GSM8K (Cobbe et al., 2021) training split and evaluate on the test split. (ii) Code Generation (**Code**): We fine-tune on CodeAlpaca (Chaudhary, 2023) and evaluate on HumanEval (Chen et al., 2021) using Pass@1/5/10.

Method	# Params (%)	GSM8K	HumanEval			HEx-PHI
			Pass@1	Pass@5	Pass@10	
LLaMA-3-8B						
FFT	100%	58.8	30.5	39.3	41.7	94.8
LoRA	1.03%	62.8	34.7	46.4	50.8	91.6
DoRA	1.05%	**63.5**	33.1	44.0	48.6	93.6
HydraLoRA	1.10%	62.8	33.4	44.7	49.3	94.2
ID-LoRA(Ours)	0.56%	**63.5**†	**37.5**	**49.7**	**55.0**	**95.3**†
Mistral-7B						
FFT	100%	55.5	29.1	38.5	40.4	94.1
LoRA	1.15%	57.8	**33.8**	42.4	45.3	91.9
DoRA	1.16%	57.5	33.7	42.6	46.8	95.3
HydraLoRA	1.22%	**58.3**	33.5	41.4	44.2	97.0
ID-LoRA (Ours)	0.62%	58.1	33.5	**43.3**†	**46.9**†	**97.2**†

Table 1: Performance comparison of different adaptation methods on single tasks, including GSM8K (Math), HumanEval (Code), and HEx-PHI (Safety) benchmarks using LLaMA-3 and Mistral. **Bold** indicates the best-performing method. † indicates statistically significant improvement, i.e., Wilcoxon signed-rank test ($p < 0.05$) against LoRA.

Method	# Params (%)	MATH	MMLU	CQA	CODE		
					Pass@1	Pass@5	Pass@10
LLaMA-3-8B							
FFT	100%	52.2	37.5	38.7	31.2	40.1	43.9
LoRA	1.03%	**60.8**	48.1	46.9	39.0	48.8	51.9
DoRA	1.05%	59.4	41.5	35.0	38.3	50.2	55.7
HydraLoRA	1.10%	60.1	47.5	44.1	33.2	43.9	47.8
MoELoRA	1.09%	53.2	41.5	35.4	28.9	41.5	47.1
VeRA	0.0178%	55.9	46.2	40.7	34.6	46.4	50.3
ID-LoRA (Ours)	**0.56%**	59.6	**51.0**	**47.4**	**41.7**	**54.5**	**58.5**
Mistral-7B							
FFT	100%	41.9	46.7	62.6	23.0	30.2	33.9
LoRA	1.15%	**52.4**	59.3	70.8	32.3	41.0	45.2
DoRA	1.16%	51.8	59.4	71.1	**32.5**	40.1	43.3
HydraLoRA	1.22%	**52.4**	60.4	71.3	32.4	41.7	45.6
MoELoRA	1.21%	44.9	60.4	71.9	29.1	38.3	41.9
ID-LoRA (Ours)	**0.62%**	52.0	**60.6**†	**74.0**	32.4	**41.8**†	**45.7**†

Table 2: Performance comparison of different adaptation methods on a multi-task suite, including Math (GSM8K), Code (HumanEval), World Knowledge (MMLU), and Common Sense (CommonsenseQA), using LLaMA-3 and Mistral. **Bold** indicates the best-performing method. "CQA" abbreviates CommonsenseQA. † indicates statistically significant improvement, i.e., Wilcoxon signed-rank test ($p < 0.05$) against LoRA.

(iii) **Safety**: We fine-tune on Saferpaca (Bianchi et al., 2023), which extends Alpaca-Cleaned (Taori et al., 2023a) with 2,000 safety instructions, and measure refusal rate on harmful queries from HEx-PHI (Qi et al., 2023).

Multi-Task. We conduct multi-task evaluation using a multi-task instruction-tuning setup: we jointly fine-tune on the concatenated training sets of Alpaca (Taori et al., 2023b), the GSM8K training split (Cobbe et al., 2021), and 50% of CodeAlpaca (Chaudhary, 2023). We evaluate multi-task capability on the GSM8K test split (Math), HumanEval (Chen et al., 2021) (Code), MMLU (Hendrycks et al., 2021) (World Knowledge), and CommonsenseQA (Talmor et al., 2019) (Common Sense).

Baselines. In single-task and multi-task experiments, we compare ID-LoRA with full fine-tuning (FFT), LoRA (Hu et al., 2022), DoRA (Liu et al., 2024), and HydraLoRA (Tian et al., 2024). In the multi-task setting, we expand the baseline pool established for single-task fine-tuning by additionally incorporating MoELoRA (Liu et al., 2023), a dedicated multi-task adaptation method. To compare with more recent parameter-efficient baselines such as Tied-LoRA (Renduchintala et al., 2024), VB-LoRA (Li et al., 2024), and VeRA (Kopiczko et al., 2024), we select the representative VeRA as the baseline, since these methods share similar motivations and design philosophies. We further include comparisons with LoRA-XS (Bałazy et al., 2024) and VeRA under the multi-task setting in Appendix B.6. **Model Configurations and Hardware**. All experiments were conducted on two NVIDIA A800 GPUs using LLaMA-3-8B (Grattafiori et al., 2024) and Mistral-7B (Jiang et al., 2023) as base models. For fair comparison, LoRA, DoRA, and HydraLoRA share identical hy-

perparameter settings with ID-LoRA. Hyperparameter details are provided in Appendix B.1.

5.2 Experiments on Single-Task

Table 1 demonstrates that ID-LoRA achieves superior parameter efficiency training only 0.56% of LLaMA-3-8B's parameters and 0.62% for Mistral-7B while consistently matching or outperforming all baselines. On the GSM8K mathematical reasoning task, ID-LoRA attains 63.5% with LLaMA-3, equaling DoRA's accuracy but with 50% fewer trainable parameters. For code generation (HumanEval), ID-LoRA delivers the highest Pass@10 scores (55.0% for LLaMA-3, 46.9% for Mistral), surpassing LoRA by 4.2–8.4% absolute gains. These results verify that ID-LoRA's low-rank decomposition preserves critical task capabilities without compromising safety.

5.3 Experiments on Multi-Task

Table 2 shows that ID-LoRA outperforms LoRA and its variants across four domains while using only half the trainable parameters (0.56% on LLaMA-3 and 0.62% on Mistral). The key driver is the clustering-guided rank allocation: it concentrates updates on task-relevant subspaces, unlocking higher effective ranks without increasing the parameter budget. Consequently, ID-LoRA leads in five of six LLaMA-3 tasks and three of six Mistral tasks, with particularly large gains on knowledge-intensive benchmarks (e.g., MMLU and CommonsenseQA). On code generation it sets new Pass@10 records, whereas on math tasks it remains on par—suggesting that global weight updates are still crucial for mathematical reasoning.

The parameter configurations in Tables 1 and 2 reveal that while HydraLoRA employs a lower rank than other methods, it exhibits a larger number of trainable parameters. For instance, in Table 2 using Mistral-7B, HydraLoRA operates at a lower rank (r=16) compared to standard LoRA (r=32), yet requires more trainable parameters (1.22% versus 1.15%). This discrepancy stems from the parameter overhead introduced by HydraLoRA's asymmetric architecture. Conversely, the proposed ID-LoRA method achieves a higher rank (r=128) while maintaining only half the trainable parameters of standard LoRA. This dual advantage primarily originates from two key innovations: (1) constructing frozen matrices through clustering of pretrained parameters, and (2) incorporating the RB algorithm.

5.4 Efficiency Analysis

Figure 3 shows the time and extra memory overhead during inference of different methods in multi-task experiments. During inference, since the adapters introduced by LoRA and DoRA can be merged with the original parameters, the model structure remains unchanged, avoiding additional memory overhead and time latency. In contrast, the adapters introduced by MoELoRA and HydraLoRA cannot be merged, resulting in extra memory usage and inference latency. The time consumption increased by 61.6% and 36.5% compared to LoRA. In our method, we only need to retain the clustering location information and the matrix B. ID-LoRA achieves a 45% reduction in extra memory usage relative to both MoELoRA and HydraLoRA, while incurring a minimal time overhead of only 0.5% compared to LoRA.

5.5 Ablation Analysis

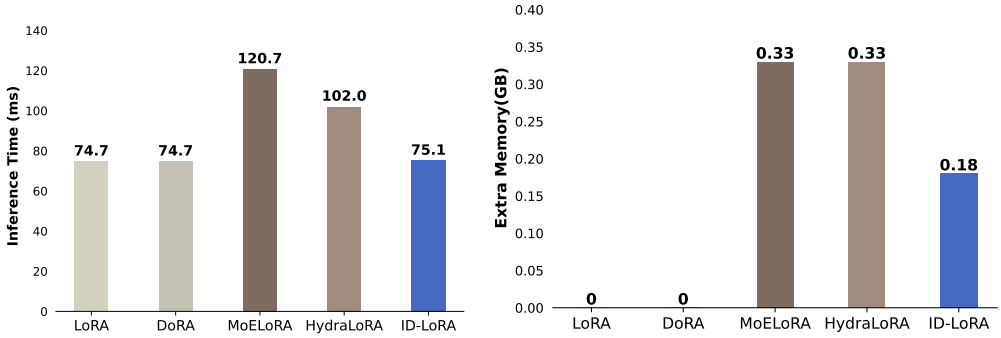
We first conducted a parameter-parity performance comparison between LoRA and ID-LoRA. Ablations then dissect the contribution of each design component: (i) PMRC Effectiveness: Comparing clustered row selection against random and serial selection strategies. (ii)RB Advantage: Evaluating rank scalability and multi-task performance under matched trainable parameter budgets.

5.5.1 Parameter-Parity Performance Analysis

Figure 4 illustrates the performance of ID-LoRA versus LoRA across four benchmarks in a multi-task setting. When both methods operate under approximately equal trainable-parameter budgets, ID-LoRA consistently surpasses LoRA, and the margin grows as the parameter ratio increases, underscoring its superior parameter efficiency and robustness. These findings confirm that, under comparable parameter constraints, ID-LoRA can employ larger ranks, leading to additional gains in multi-task modeling.

5.5.2 Experiments on PMRC and RB

To validate PMRC's clustering-based grouping, we compare it with two baselines on LLaMA-3-8B under a multi-task setup: Serial Selection (SS) takes contiguous rows and Random Selection (RS) samples rows uniformly from the frozen pretrained weight matrix. Table 3 contrasts PMRC-based grouping with two baselines, Serial (SS+RB) and Random (RS+RB) row selection, on LLaMA-3-8B. With equal parameter budgets, PMRC consis-



(a) Inference Time

(b) Extra GPU Memory

Figure 3: The inference time and extra memory overhead of different adaptation methods under the same hyperparameter settings as the multi-task experiments on LLaMA-3-8B, tested on an A800 GPU.

Method	# Params (%)	MATH	MMLU	CQA	CODE		
					Pass@1	Pass@5	Pass@10
LLaMA-3-8B							
LoRA	1.03%	60.8	48.1	46.9	39.0	48.8	51.9
ID-LoRA (w/o RB)	0.60%	59.4	50.2	46.4	40.2	50.8	53.7
ID-LoRA (SS+RB)	0.56%	58.8	50.7	47.7	40.7	53.5	56.2
ID-LoRA (RS+RB)	0.56%	59.2	51.0	47.2	40.1	52.8	55.4
ID-LoRA (PMRC+RB)	**0.56%**	59.6	51.0	47.4	41.7	54.5	58.5
ID-LoRA (Random Init.)	0.56%	60.2	49.2	44.9	40.3	53.4	55.2
ID-LoRA (SVD Init.)	0.56%	47.9	35.2	44.6	33.7	42.3	45.3

Table 3: Ablation on LLaMA-3-8B across a multi-task dataset comparing different strategies and initialization methods. 'w/o RB' denotes removing the RB algorithm from ID-LoRA. 'SS' abbreviates Serial Selection, where pivots are assigned to groups in sequential order. 'RS' abbreviates Random Selection, where pivots are assigned to groups at random. 'Random Init.' and 'SVD Init.' replace MID-based bases with randomly initialized or SVD-derived bases, while keeping the same trainable-parameter budget.

tently outperforms both baselines across all tasks, confirming that clustering-based initialization enables superior expert specialization and downstream gains.

Table 3 presents a controlled ablation on the LLaMA-3-8B multi-task benchmark to isolate the contribution of RB (PMRC+RB vs w/o RB). We report the corresponding rank configurations and trainable-parameter budgets for this comparison in Appendix B.5. Across four multi-task evaluation benchmarks, the RB-enhanced variant delivers consistent gains, except on MATH. MATH tasks necessitate precise and accurate directional updates to the weight matrix; simply increasing the rank through the RB algorithm alone is insufficient to enhance performance. These results validate the intrinsic effectiveness of the RB algorithm in multi-task scenarios. Notably, RB achieves these gains under approximately matched trainable-parameter budgets, indicating that the improvements stem from higher effective rank rather than additional trainable parameters.

6 Conclusion

We propose ID-LoRA, a parameter-efficient fine-tuning framework that reuses pretrained weight structure via matrix interpolative decomposition. To further improve multi-task modeling under strict parameter budgets, we introduce Rank Boosting (RB), which boosts the effective rank via feature splitting and concatenation without increasing trainable parameters.

We provide theoretical support showing that when tasks exhibit a clustered structure, cluster-aware decomposition yields tighter approximation guarantees than global low-rank adaptation. Experiments on single-task and multi-task benchmarks further demonstrate that ID-LoRA achieves competitive or improved performance while using substantially fewer trainable parameters than standard LoRA and recent PEFT baselines. Future work will explore more robust grouping and composition strategies that improve performance on challenging reasoning regimes while preserving ID-LoRA's efficiency benefits.

7 Limitations

ID-LoRA relies on the assumption that pretrained weight matrices contain reusable directions that are close to the low-rank subspaces needed for downstream tasks. When target tasks are highly out-of-distribution relative to pretraining, or when adaptation requires sharp task-specific directions, reusing frozen bases may provide limited gains. In our experiments, improvements are less pronounced on mathematical reasoning, which often demands more precise directional updates than rank expansion alone can provide.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328, Online. Association for Computational Linguistics.

Meta AI. 2024. The llama 3.2 herd of models. https://huggingface.co/meta-llama/Llama-3.2-3B. Llama-3.2-3B variant used in our experiments.

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. 2024. Lora-xs: Low-rank adaptation with extremely small number of parameters.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Röttger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. 2023. Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. *arXiv preprint arXiv:2309.07875*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. GitHub repository. Accessed: 2025-07-04.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5:220–235.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Meng Fang, Shilong Deng, Yudi Zhang, Zijing Shi, Ling Chen, Mykola Pechenizkiy, and Jun Wang. 2024. Large language models are neurosymbolic reasoners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17985–17993.

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

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Kenneth L. Ho and Leslie Greengard. 2012. A fast direct solver for structured linear systems by recursive skeletonization. 34:A2507–A2532.

Roger A. Horn and Charles R. Johnson. 2012. *Matrix Analysis*, 2nd edition. Cambridge University Press, Cambridge. Includes over 1,100 problems and new sections on SVD, CS decomposition, and Weyr canonical form.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2023. Lorahub: Efficient cross-task generalization via dynamic lora composition. *arXiv preprint arXiv:2307.13269*.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

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. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2024. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*.

Josh Levy-Kramer. 2018. k-means-constrained.

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

Yang Li, Shaobo Han, and Shihao Ji. 2024. Vb-lora: Extreme parameter efficient fine-tuning with vector banks.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. 2023. Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications. *CoRR*, abs/2310.18339.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, pages 32100–32121.

R. Penrose and J. A. Todd. 1956. On best approximate solutions of linear matrix equations. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52(01):17.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. 2023. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*. This paper contains red-teaming data and model-generated content that can be offensive in nature.

Yanyuan Qiao, Zheng Yu, and Qi Wu. 2023. Vln-petl: Parameter-efficient transfer learning for vision-and-language navigation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15397–15406.

Adithya Renduchintala, Tugrul Konuk, and Oleksii Kuchaiev. 2024. Tied-lora: Enhancing parameter efficiency of lora with weight tying.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023a. Stanford alpaca: An instruction-following llama model. GitHub repository. Accessed: 2025-07-04.

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

Qwen Team. 2025. Qwen3 technical report.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Cheng-Zhong Xu. 2024. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *Advances in Neural Information Processing Systems*, 37:9565–9584.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, and Jian Yang. 2024. Qwen2 technical report.

Juzheng Zhang, Jiacheng You, Ashwinee Panda, and Tom Goldstein. 2025. Lori: Reducing cross-task interference in multi-task low-rank adaptation.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. 2023. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning. *arXiv preprint arXiv:2308.03303*. Version: 2023b.

A Theoretical Framework of ID-LoRA

To preserve theoretical completeness, we first restate the three assumptions from the paper, then, derive Theorems 1 and 2 via the detailed definitions that follow, providing full proofs in this section.

A.1 Problem Setup and Assumptions

Assumption 3 (Task Clustering Structure) *Let $\{W_i^*\}_{i=1}^m \subset \mathbb{R}^{d_1 \times d_2}$ denote the optimal parameter matrices for m related tasks. We assume these matrices can be partitioned into k disjoint clusters $\mathcal{C}_1, \mathcal{C}_2 \ldots, \mathcal{C}_k$ (with $k < m$) such that:*

$$\forall i, j \in \mathcal{C}_l, \quad \textit{Rank}(W_i^* - W_j^*) \leq r_l < \min(d_1, d_2) \tag{11}$$

where $\mathcal{C}_l$ denotes the l-th cluster, and r_l is the intrinsic rank difference within the cluster.

Assumption 4 (Task Parameter Sharing) *We assume the existence of a shared feature matrix $B \in \mathbb{R}^{d_2 \times r_g}$ with $r_g < d_2$, such that every optimal task-specific parameter matrix can be written as:*

$$W_i^* = BA_i + E_i, \quad \|E_i\|_F^2 \leq \epsilon \tag{12}$$

where $A_i \in \mathbb{R}^{r_g \times d_1}$ is a task-specific matrix and E_i is the reconstruction error.

Definition 2 (Pivot Sensitivity) *The* CUR *decomposition's approximation accuracy crucially depends on pivot selection, with reconstruction errors escalating sharply when the pivots inadequately capture the principal row or column subspaces,*

$$\max_{\textit{bad pivots}} \|W - \tilde{W}_{\text{CUR}}\|_F^2 > \min_{\textit{good pivots}} \|W - \tilde{W}_{\text{CUR}}\|_F^2 \tag{13}$$

A.2 Core Definitions

Definition 3 (Task Parameter Distance) *For any pair of tasks $i, j \in [m]$, their distance is defined as the squared Frobenius norm of the difference between their optimal parameter matrices:*

$$d(i, j) = \|W_i^* - W_j^*\|_F^2 \tag{14}$$

Definition 4 (Cluster Low-Rank Decomposition) *Let the optimal parameter matrices of the m tasks be $\{W_i^*\}_{i=1}^m \in \mathbb{R}^{d_1 \times d_2}$.*

(1) Perform k-means on $\{W_i^*\}_{i=1}^m$ to obtain disjoint clusters $\{\mathcal{C}_l\}_{l=1}^k$.

(2) For each cluster $\mathcal{C}_l$, compute the centroid matrix $M_l = \frac{1}{|\mathcal{C}_l|} \sum_{i \in \mathcal{C}_l} W_i^*$ and extract its rank r_l truncated SVD, yielding the cluster-specific matrix $A_l \in \mathbb{R}^{d_1 \times r_l}$.

(3) *Learn the shared matrix $B \in \mathbb{R}^{d_2 \times r}$ by solving*

$$\min_B \sum_{i=1}^{m} \|W_i^* - BA_{l(i)}\|_F^2, \tag{15}$$

where $l(i)$ denotes the cluster of task i.

Definition 5 (Multi-Task Reconstruction) *The parameter matrix for task i is reconstructed as*

$$\tilde{W}_i = BA_{l(i)} \tag{16}$$

Definition 6 (CUR Decomposition) *Given a matrix $W \in \mathbb{R}^{d \times d}$, a* CUR *decomposition approximates W as a product of three matrices:*

$$W \approx CUR \tag{17}$$

where $C \in \mathbb{R}^{d \times c}$, and is a subset of c columns from W. $R \in \mathbb{R}^{r \times n}$ and is a subset of r rows from W. $U \in \mathbb{R}^{c \times r}$, which is a weight matrix constructed to minimize the approximation error $\|A - CUR\|_F$.

A.3 Main Theorems

Theorem 3 (Clustering Reconstruction) *Under Assumptions 1 and 2, the clustering-aware decomposition achieves a tighter reconstruction error bound compared to global low-rank decomposition:*

$$\sum_{i=1}^{m} \|W_i^* - \tilde{W}_i\|_F^2 \leq \sum_{i=1}^{m} \|W_i^* - \tilde{W}_i^{global}\|_F^2 - \Delta \tag{18}$$

where $\Delta = \sum_{l=1}^{k} \sum_{i \in \mathcal{C}_l} \|B(A_{l(i)} - A^{global})\|_F^2 \geq 0$. The inequality becomes strict ($\Delta > 0$) when tasks exhibit clustering structure.

Proof. Let $\tilde{W}_i^{\text{global}} = BA^{\text{global}}$ denote the reconstruction obtained from the globally shared low-rank factorization. By **Assumption 2**, we have

$$\left\|W_i^* - \tilde{W}_i^{\text{global}}\right\|_F^2 = \left\|B(A_i - A^{\text{global}}) + E_i\right\|_F^2. \tag{19}$$

In addition, the cluster-aware reconstruction (i.e., **Definition 2**) satisfies

$$\left\|W_i^* - \tilde{W}_i\right\|_F^2 = \left\|B(A_i - A_{l(i)}) + E_i\right\|_F^2. \tag{20}$$

Since $A_{l(i)}$ is the cluster centroid, we have

$$\|A_i - A_{l(i)}\|_F^2 \leq \|A_i - A^{\text{global}}\|_F^2, \tag{21}$$

and B is optimized to minimize the within-cluster approximation error. Consequently, $\Delta \geq 0$. Whenever the inter-cluster task discrepancy is significant, $\Delta > 0$, demonstrating the superiority of the cluster-aware decomposition.

Theorem 4 (Cluster-Pivot Stability) *Under Definition 1, the clustering-based local pivot selection ensures a tighter expected error bound compared to global pivot selection:*

$$\mathbb{E}_{\mathcal{P}}\left[\max_{1 \leq i \leq m} \|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F^2\right] \geq \mathbb{E}_{\mathcal{P}_l}\left[\max_{1 \leq i \leq m} \|W_i^* - \tilde{W}_i\|_F^2\right], \tag{22}$$

where $\tilde{W}_i^{\text{CUR}}$ is the reconstruction parameter using global pivots on all tasks, $\tilde{W}_i$ is the reconstruction parameter using cluster-local pivots (per-cluster CUR *on $\mathcal{C}_l$), and $\mathcal{P}_l$ is the set of pivots selected from cluster $\mathcal{C}_l$'s centroid matrix.*

Proof. Step 1: Error Decomposition and Notation.

(1) *Global Approximation Error*: For task i, the global CUR approximation is:

$$\tilde{W}_i^{\text{CUR}} = C_{\mathcal{P}} U_{\mathcal{P}} R_{\mathcal{P}}, \quad \mathcal{P} \subseteq [d_1] \times [d_2], |\mathcal{P}| = s \tag{23}$$

where pivot set $\mathcal{P}$ is uniformly sampled from all rows or columns.

(2) *Local Approximation Error*: For task i in cluster $\mathcal{C}_l$:

$$\tilde{W}_i = C_{\mathcal{P}_l} U_{\mathcal{P}_l} R_{\mathcal{P}_l}, \mathcal{P}_l \sim \text{cluster-specific distribution} \tag{24}$$

with pivots $\mathcal{P}_l$ selected from cluster-specific rows or columns.

Step 2: Lower Bound for Global Error.

By Definition 1, there exists at least one cluster $\mathcal{C}_{l^*}$.

(1) Rare Feature Non-Ignorability:

$$\mathbb{P}_{\mathcal{P}}(\mathbf{v}_{l^*} \notin \text{span}(C_{\mathcal{P}})) \geq \eta_{l^*} > 0 \tag{25}$$

where $\mathbf{v}_{l^*}$ is the unique feature direction of $\mathcal{C}_l$, $\|\mathbf{v}_{l^*}\|_2 = 1$. Since it has negligible weight in the global feature space (e.g., $\sigma_{l^*} < \sum \sigma_i$).

(2) Error Lower Bound: when $\mathbf{v}_{l^*}$ is missed:

$$\|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F \geq \gamma_{l^*} \quad \forall i \in \mathcal{C}_{l^*} \tag{26}$$

Thus:

$$\mathbb{E}_{\mathcal{P}}\left[\max_i \|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F\right] \geq \gamma_{l^*} \cdot \eta_{l^*} \tag{27}$$

where $\gamma_{l^*} = \sup_{\mathcal{P}: \mathbf{v}_{l^*} \notin \text{span}(C_{\mathcal{P}})} \|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F$ (minimum error when $\mathbf{v}_{l^*}$ is missed globally).

Step 3: Upper Bound for Local Error.

For any cluster $\mathcal{C}_l$, local pivot selection satisfies:

(1) Feature Coverage Property:

$$\mathbb{P}_{\mathcal{P}_l}(\mathbf{v}_l \in \text{span}(C_{\mathcal{P}_l})) \geq 1 - \epsilon_l \quad (\epsilon_l \leq e^{-c|\mathcal{P}_l|}) \tag{28}$$

because $\mathbf{v}_l$ is prominent within $\mathcal{C}_l$ (by Assumption 1).

(2) Error Decomposition:

$$\|W_i^* - \tilde{W}_i\|_F \leq \underbrace{\|M_l - \tilde{M}_l\|_F}_{\text{cluster center error}} + \underbrace{\|W_i^* - M_l\|_F}_{\leq \delta_l} \tag{29}$$

where $\tilde{M}_l = C_{\mathcal{P}_l} U_{\mathcal{P}_l} R_{\mathcal{P}_l}$.

(3) Low-Rank Approximation: By Assumption 1 ($\text{rank}(M_l) \leq r_l$) and CUR theory:

$$\|M_l - \tilde{M}_l\|_F \leq (1 + \sqrt{r_l})\|M_l - M_l^{(r_l)}\|_F = 0 \tag{30}$$

when $\mathbf{v}_l \in \text{span}(C_{\mathcal{P}_l})$. Hence:

$$\mathcal{E}_l^{\text{local}} \leq \delta_l \quad \text{with probability} \geq 1 - \epsilon_l \tag{31}$$

where $\delta_l = \max_{i \in \mathcal{C}_l} \|W_i^* - M_l\|_F$ (intra-cluster consistency measure).

(4) Expectation Control:

$$\mathbb{E}[\mathcal{E}_l^{\text{local}}] \leq \delta_l(1-\epsilon_l) + (\text{diam}(\mathcal{C}_l) + \|M_l\|_F)\epsilon_l \leq \delta_l + \mathcal{O}(\epsilon_l) \tag{32}$$

where $\text{diam}(\mathcal{C}_l) = \max_{i,j \in \mathcal{C}_l} \|W_i^* - W_j^*\|_F$.

Step 4: Global vs. Local Error Comparison:

$$\mathbb{E}[\mathcal{E}_{\text{global}}] \geq \gamma_{l^*}\eta_{l^*}, \quad \mathbb{E}[\mathcal{E}_{\text{local}}] \leq \max_l(\delta_l + \mathcal{O}(\epsilon_l)) \tag{33}$$

From Definition 1 ($\gamma_{l^*}\eta_{l^*} > \delta_l + \epsilon_l$), the theorem follows. First, global error lower bound depends on rare feature miss probability η_{l^*} and minimum miss error γ_{l^*}. Then, local error upper bound is controlled by intra-cluster consistency δ_l and feature coverage failure ϵ_l.

B Experimental Details and Additional Results

This appendix provides comprehensive experimental details omitted from the main text due to space constraints, along with additional results that further validate the effectiveness of our method. We first restate all hyperparameter settings, and computational-resource specifications to ensure full reproducibility. We then conduct extended ablation studies on key design choices, examining how the selection of the cluster number k influences multi-task performance, and we include additional multi-task evaluations that were omitted from the main paper. Table 6 presents the number of trainable parameters under identical rank settings for different methods on the LLaMA 3-8B model. These results show that ID-LoRA requires the fewest trainable parameters among baseline methods under the same rank.

B.1 Hyperparameter Settings

In this section, we provide detailed hyperparameter settings for the main experiments in the paper, which are presented in Tables 4 and 5. Table 4 details the fine-tuning configurations of the ID-LoRA method in single-task settings for Math, Code, and Safety tasks, applied to two base models: LLaMA-3-8B and Mistral-7B. Table 5 shows the hyperparameter settings for multi-task scenarios.

B.2 Experimental Details of Parameter-Parity Performance

The results in Table 7 provide the detailed numerical data for Figure 4, which could not be fully presented in the main text due to space constraints. As the proportion of trainable parameters rises from 0.13% to 1%, both methods improve on every task. ID-LoRA already surpasses LoRA at 0.56% parameters, attaining 59.6 on MATH and 41.7 Pass@1 on HumanEval, and achieves the overall best performance at 0.99% parameters (MATH 60.7, MMLU 51.8, HumanEval Pass@1 43.4).

B.3 Ablation: Impact of Pretrained Parameter Cluster Number k

To ablate the effect of the cluster count on model behavior, we conducted multi-task experiments on LLaMA-3.2-3B with ranks fixed at r=128 and cluster numbers set to $k = 2, 4, 6, 8$. The outcomes, reported in Table 8, guided our choice of $k = 4$ (or $k = 6$) for the main experiments.

As the cluster number k increases, the amount of copied parameters in the pretrained parameter matrix also rises accordingly, which reduces the distinction between different sub-matrices A_i, weakening the diverse combinatorial advantages brought by the routing mechanism. The performance decline at $k = 8$ reflects this issue of diminished expressive power caused by excessive replication. Due to computational constraints, this sensitivity study is conducted on LLaMA-3.2-3B; core component ablations on LLaMA-3-8B are reported in Table 3.

B.4 Ablation: Impact of Initialization Strategies

We evaluated Matrix Interpolative Decomposition (MID) against SVD and Random Initialization. Results in Table 9 show that MID (ID-LoRA) consistently outperforms both. SVD performs significantly worse (e.g., 35.2 vs 51.0 on MMLU),

Hyperparameter	GSM8K		CodeAlpaca		Saferpaca	
	LLaMA-3	Mistral	LLaMA-3	Mistral	LLaMA-3	Mistral
Rank/α	128 / 256	128 / 256	128 / 256	128 / 256	128 / 256	128 / 256
Learning Rate	5e-5	3e-5	5e-5	5e-5	1e-4	1e-4
Dropout	0.05	0.05	0.05	0.05	0.05	0.05
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Batch Size	32	32	32	32	32	32
Warmup Steps	0	0	0	0	0	0
Epochs	3	3	2	2	2	2

Table 4: Hyperparameter settings for ID-LoRA on GSM8K, CodeAlpaca, and Saferpaca datasets. The ID-LoRA parameters are placed at the q, k, v, o, gate, up, and down positions.

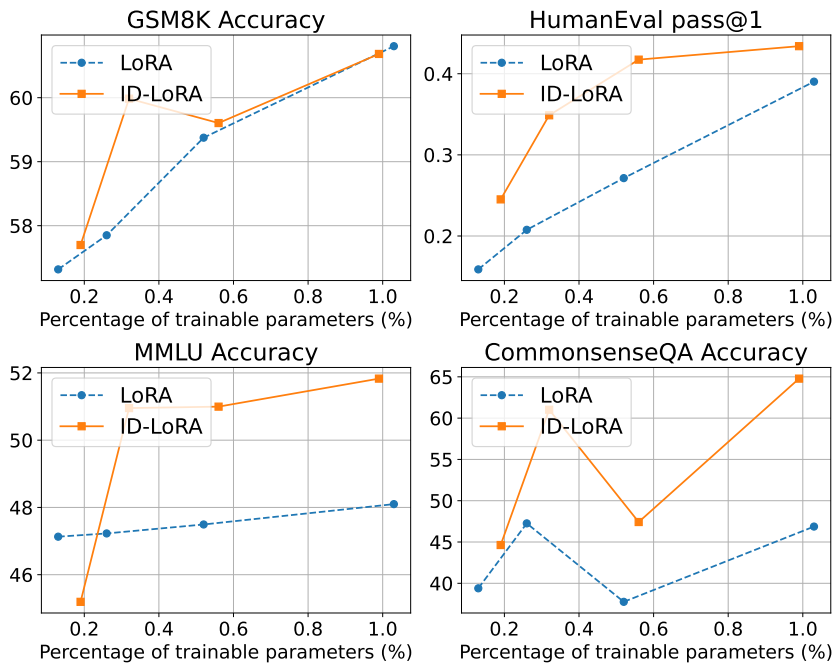


Figure 4: Performance comparison between ID-LoRA and vanilla LoRA on four representative benchmarks (GSM8K, HumanEval, MMLU, and CommonsenseQA) using LLaMA-3-8B as the backbone. Both methods vary the rank while keeping the number of trainable parameters approximately equal, and results are reported under few-shot or zero-shot settings. Detailed results are provided in Appendix B.2.

Method	ID-LoRA	ID-LoRA
Base Model	LLaMA-3	Mistral
Rank r	128	128
α	256	256
Learning Rate	5e-5	5e-5
Dropout	0.05	
Optimizer	AdamW	
Batch size	32	
Warmup Steps	0	
Epochs	2	
Where	q, k, v, o, gate, up, down	

Table 5: Hyperparameter settings for ID-LoRA on the multi-task dataset

likely because its global principal components fail to capture the diverse local structures required for multi-task adaptation. While Random Initialization is competitive, it lags behind MID on complex reasoning tasks (MMLU, CQA, Code), demonstrating that reusing pretrained structures via MID provides a critical initialization advantage for multi-task learning.

B.5 Ablation: Interpreting RB under Parameter-Parity Budgets

The RB ablation in Table 3 compares two variants under similar trainable-parameter budgets. The apparent counterintuitiveness stems from conflating *effective rank* with *trainable parameters*: without RB, increasing rank directly enlarges the shared trainable matrix B (and thus the trainable-parameter count), so the w/o RB variant is constrained to a smaller rank ($r = 32$). RB reduces the parameterization of B via feature splitting and shared projections, enabling a higher effective rank ($r = 128$) under a comparable budget. Conse-

	Rank	LoRA # parameters	DoRA # parameters	HydraLoRA # parameters	MoELoRA # parameters	ID-LoRA # parameters
LLAMA3-8B	8	21.0M	22.3M	46.7M	88.9M	7.7M
	16	41.9M	43.3M	89.7M	172.8M	10.4M
	32	83.9M	85.3M	175.7M	340.5M	16.0M
	64	167.8M	169.1M	347.7M	676.1M	27.0M
	128	335.5M	336.9M	691.6M	1.3B	49.0M

Table 6: Number of trainable parameters for different methods on LLaMA-3-8B

Method	**# Params (%)**	**MATH**	**MMLU**	**CQA**	**CODE**		
					Pass@1	**Pass@5**	**Pass@10**
LoRA	0.13 (r=4)	57.3	47.1	39.4	15.9	24.6	29.1
	0.26 (r=8)	57.9	47.2	47.3	20.7	30.3	34.5
	0.52 (r=16)	59.4	47.5	37.8	27.1	37.3	42.2
	1.03 (r=32)	60.8	48.1	46.9	39.0	48.8	51.9
ID-LoRA	0.19 (r=32)	57.7	45.2	44.6	24.5	32.7	36.2
	0.32 (r=64)	60.0	51.0	61.0	34.9	47.7	53.7
	0.56 (r=128)	59.6	51.0	47.4	41.7	54.5	58.5
	0.99 (r=256)	60.7	51.8	64.8	43.4	55.9	59.9

Table 7: Performance comparison of LoRA and ID-LoRA on the LLaMA-3-8B backbone under multi-task instruction-tuning. All methods are trained on the same mixture (Alpaca + GSM8K + 50% CodeAlpaca) and evaluated on MATH, MMLU, CommonsenseQA (CQA), and HumanEval (Pass@1/5/10). These numbers correspond to the experiment shown in Figure 4 of the paper.

k	**MATH**	**MMLU**	**COM**	**CODE**		
				Pass@1	**Pass@5**	**Pass@10**
2	38.3	**48.7**	64.4	29.0	34.7	37.1
4	38.1	48.5	**65.6**	29.4	35.9	38.2
6	**39.6**	46.9	65.1	**30.6**	**36.5**	**38.4**
8	38.1	45.5	64.7	30.2	36.0	38.2

Table 8: Ablation on the number of pretrained parameter clusters k under the multi-task setting. Results are obtained with the LLaMA-3.2-3B (AI, 2024) backbone. The table reports overall multi-task accuracy for each choice of k.

quently, the RB variant can be stronger even though it does not rely on more trainable parameters.

B.6 Additional Baselines: LoRA-XS and VeRA under Multi-Task Instruction-Tuning

Table 11 shows that ID-LoRA outperforms both LoRA-XS and VeRA on all evaluated multi-task benchmarks. Although LoRA-XS and VeRA achieve extremely low trainable-parameter counts, their performance remains substantially below ID-LoRA on knowledge-intensive and generation tasks (MMLU and HumanEval). These results support the central claim that reusing pretrained structures via MID and enabling higher effective ranks through RB yields stronger multi-task adaptation under practical parameter budgets.

B.7 Additional Results on a Newly Released Backbone: Qwen3-8B

Table 12 demonstrates that ID-LoRA remains effective on a newly released base model. Compared with LoRA, ID-LoRA achieves higher accuracy on MMLU (+3.1), CommonsenseQA (+11.6), and CODE generation (+3.2 Pass@1), while training roughly half of the parameters (0.56% vs 1.05%). The slightly lower MATH score is consistent with our observation that mathematical reasoning often requires more precise task-specific weight updates than rank expansion alone provides, suggesting that incorporating stronger directionally constrained updates is a promising future direction.

B.8 Additional Baseline on Qwen3-8B: LoRI-D

Although LoRI is designed for multi-task low-rank adaptation, its formulation emphasizes per-task training and adapter merging, which differs from ID-LoRA's single-stage training on mixed data with a router. Following the reviewer suggestion, we additionally compare against LoRI-D (Zhang et al., 2025) under the same mixed-data training setup. As shown in Table 13, ID-LoRA consistently outperforms LoRI-D across all tasks. We attribute this advantage to ID-LoRA's trainable router, which learns input-dependent expert com-

Method	MATH	MMLU	CQA	CODE		
				Pass@1	Pass@5	Pass@10
LLaMA-3-8B						
Random Init.	60.2	49.2	44.9	40.3	53.4	55.2
SVD Init.	47.9	35.2	44.6	33.7	42.3	45.3
ID-LoRA (Ours)	59.6	51.0	47.4	41.7	54.5	58.5

Table 9: Performance comparison of different initialization strategies on multi-task benchmarks using LLaMA-3-8B.

Method	# Params (%)	Rank r
ID-LoRA (PMRC+RB)	0.56%	128
ID-LoRA (w/o RB)	0.60%	32

Table 10: Rank configurations and trainable-parameter budgets for the RB ablation on LLaMA-3-8B in Table 3.

position, whereas LoRI-D relies on a linear combination governed by manually specified hyperparameters, limiting its ability to adaptively resolve cross-task interference.

B.9 Router Activation Statistics

To assess whether the router weights can be replaced by fixed mixture coefficients, we analyze the router activation values during inference. We sample 100 examples from each of the CODE, MATH, and CQA test sets, and collect the router activation values $\{\alpha_i\}_{i=1}^{4}$ from the `self_attn.q_proj` layer at the 20th decoder layer when generating the first token in the reasoning process. We report the mean and standard deviation (Std Dev) in Table 14.

Statistic	Router 0	Router 1	Router 2	Router 3
Mean	0.2026	0.1777	0.1968	0.4220
Std Dev	0.0924	0.0350	0.0439	0.1097

Task / Statistic	Router 0	Router 1	Router 2	Router 3
CODE Mean	0.2646	0.1606	0.2181	0.3566
CODE Std Dev	0.0325	0.0245	0.0360	0.0490
MATH Mean	0.1019	0.2049	0.1585	0.5347
MATH Std Dev	0.0142	0.0287	0.0170	0.0476
CQA Mean	0.3197	0.1469	0.2485	0.2849
CQA Std Dev	0.0204	0.0155	0.0180	0.0252

Table 14: Router activation statistics. Top: aggregated across CODE, MATH, and CQA. Bottom: statistics by task type.

Table 14 shows that Router 3 has the highest average activation overall, but the by-task statistics reveal clear task-dependent patterns: MATH strongly favors Router 3 (0.5347) while suppressing Router 0 (0.1019), whereas CQA shifts probability mass toward Router 0 (0.3197) and Router 2 (0.2485), with a substantially lower Router 3 (0.2849). The relatively small within-task standard deviations indicate that these preferences are consistent across sampled instances. These results suggest that using a fixed set of mixture coefficients would fail to capture task-dependent routing behavior; therefore, learnable and input-dependent router weights are necessary for robust multi-task adaptation.

Method	# Params (%)	MATH	MMLU	CQA	CODE		
					Pass@1	Pass@5	Pass@10
LLaMA-3-8B							
LoRA-XS (r=32)	0.0028%	54.6	44.7	37.7	31.9	45.5	48.9
LoRA-XS (r=256)	0.1684%	56.7	46.8	39.9	35.5	47.1	50.8
VeRA (r=32)	0.0172%	55.2	46.0	40.1	34.8	46.8	50.6
VeRA (r=256)	0.0178%	55.9	46.2	40.7	34.6	46.4	50.3
ID-LoRA (r=128)	0.56%	59.6	51.0	47.4	41.7	54.5	58.5

Table 11: Comparison with LoRA-XS and VeRA on the LLaMA-3-8B backbone under the multi-task instruction-tuning setup.

Method	# Params (%)	MATH	MMLU	CQA	CODE		
					Pass@1	Pass@5	Pass@10
Qwen3-8B							
LoRA	1.05%	84.6	45.4	67.2	62.9	73.5	77.0
ID-LoRA (Ours)	0.56%	82.7	48.5	78.8	66.1	77.2	79.6

Table 12: Multi-task instruction-tuning results on the newly released Qwen3-8B backbone (Team, 2025).

Method	# Params (%)	MATH	MMLU	CQA	CODE		
					Pass@1	Pass@5	Pass@10
Qwen3-8B							
LoRI-D (Zhang et al., 2025)	0.54%	81.9	44.7	72.8	63.6	75.7	78.2
ID-LoRA (Ours)	0.56%	82.7	48.5	78.8	66.1	77.2	79.6

Table 13: Comparison with LoRI-D on Qwen3-8B under the same mixed-data multi-task training setup.