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

Anonymous EMNLP2026 submission

Abstract

LoRA is a widely used PEFT technique for LLMs, but scaling it to large models incurs substantial trainable-parameter overhead. Lowering rank curbs cost yet hurts multi-task performance. We propose ID-LoRA, which clusters pretrained weight rows into frozen bases $\{A_i\}$, composes them through one shared matrix B, and widens the composed update via rank boosting (RB). On five benchmarks, ID-LoRA matches or exceeds strong baselines on LLaMA-3-8B with up to 46% fewer parameters than standard LoRA, with the largest gains on code and knowledge tasks. In multi-task settings, GSM8K sometimes trails LoRA (59.6 vs. 60.8), while on Mistral most metrics stay within ≤ 0.4 points of the best baseline.

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. Recent multi-LoRA and router-based PEFT methods (Huang et al., 2023; Liu et al., 2023; Tian et al., 2024) address heterogeneity by maintaining several adapters, but typically still train multiple (A_i, B_i) pairs at $\mathcal{O}(krd)$ cost. ID-LoRA instead freezes clustered pretrained rows as $\{A_i\}$, trains one shared B, and routes over experts—closer in spirit to mixture-style composition, yet with far fewer trainable weights (Section 2).

To address this limitation, we propose ID-LoRA, a low-rank PEFT framework inspired by Matrix Interpolative Decomposition (MID) (Ho and Greengard, 2012). The design is guided by two structural hypotheses about multi-task adaptation: related tasks admit a *clustering structure*—their optimal updates occupy a few disjoint low-rank subspaces rather than one global subspace—and, within each subspace, a *shared low-dimensional basis* can compose task-specific adapters with bounded error. Section 3.3 formalizes these assumptions and shows that cluster-aware CUR-style pivot selection yields tighter reconstruction than a single global low-rank adapter when tasks differ across clusters.

ID-LoRA turns these hypotheses into a four-module pipeline (Figure 1 (b)): **Parameter Matrix Row Clustering (PMRC)** partitions rows of frozen $W \in \mathbb{R}^{d\times d}$ into k groups via constrained k-means (Levy-Kramer, 2018) and selects the r rows nearest each centroid as frozen bases $\{A_i\}$; **shared-basis adaptation** trains one matrix B to compose all $\{A_i\}$; **input-dependent routing** mixes experts via $\alpha_i = \mathrm{softmax}_i(\langle T, A_i h_t\rangle)$; and **Rank Boosting (RB)** widens the composed update dimension via feature splitting and concatenation without proportionally increasing trainable parameters (Section 2 defines nominal rank and composed update dimension).

We validate these hypotheses via multi-task benchmarks, parameter-parity sweeps, PMRC/RB ablations (Tables 10, 9), and router statistics (Appendix B.11); Section 4.6 and Table 11 tie each claim to evidence. Under matched budgets, ID-LoRA is strong on code and safety, competitive on single-task math, and below LoRA on multi-task math (59.6 vs. 60.8) while using 46% fewer trainable parameters and gaining 2.9 MMLU points. The main contributions are: (1) ID-LoRA—frozen clustered rows, shared B, routing, and RB—widens the composed update dimension with far

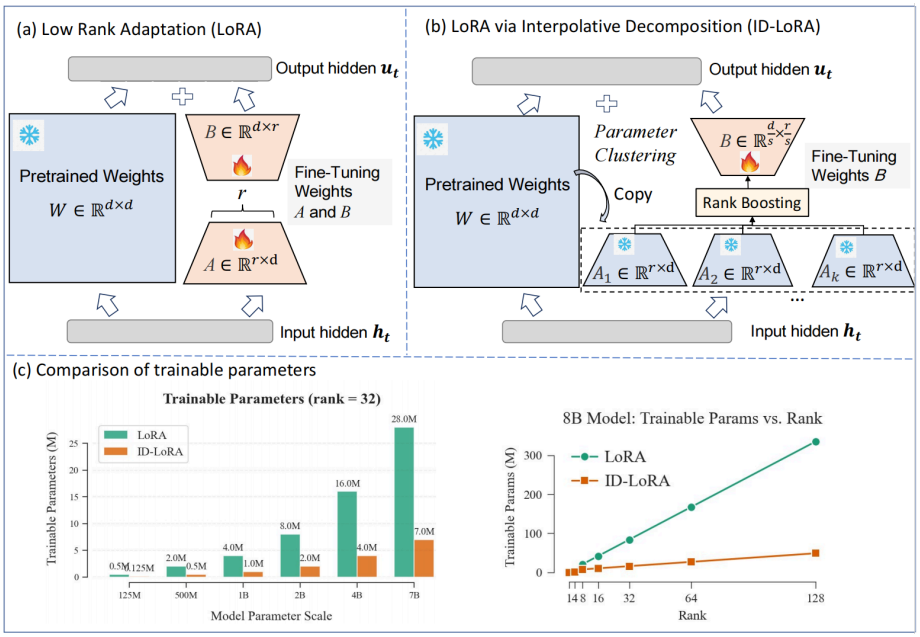


Figure 1: Architectural comparison: LoRA versus ID-LoRA. (a) LoRA trains A and B. (b) ID-LoRA clusters frozen rows into $\{A_i\}$, shares one B, and applies RB. (c) Trainable-parameter count versus model size (left) and rank (right): ID-LoRA uses $\sim$5$\times$ fewer trainable parameters than LoRA at r=32 in our setting; accuracy gains remain task-dependent (Section 4.3).

fewer trainable weights than router-based multi-LoRA baselines, supported by cluster-aware CUR-style theory and component ablations. (2) Empirical evidence on LLaMA-3-8B and Mistral-7B with in-text guidance on preprocessing, training/inference cost, and choosing k and r (Section 4.2), acknowledging task-dependent and near-parity results on math and some Mistral-7B metrics. (3) A transparent claim–evidence map (Table 11) linking Assumptions 1–2, Theorems 1–2, and module ablations, including settings where LoRA remains preferable.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT). PEFT adapts LLMs with minimal parameter changes (Houlsby et al., 2019; Hu et al., 2022). Representative approaches include Adapter (Houlsby et al., 2019), Prefix-tuning (Li and Liang, 2021), and LoRA (Hu et al., 2022), which decompose weight updates as low-rank products but face a rank–parameter trade-off (Zhang et al., 2023, 2025). Recent variants (Liu et al., 2024; Kopiczko et al., 2024) improve capacity or efficiency yet still struggle in heterogeneous multi-task settings.

Multi-LoRA and router-based PEFT. Multi-task adaptation maintains multiple adapters instead of one global LoRA. Our main baselines—LoRA, DoRA, HydraLoRA, and **MOELoRA** (Liu et al., 2023) (Liu et al., 2023)—are widely adopted for fair comparison; Appendix runs additionally include VeRA (Kopiczko et al., 2024), LoRA-XS (Bałazy et al., 2024), and LoRI-D (Zhang et al., 2025). **LoRAHub** (Huang et al., 2023) composes offline-trained adapters at inference; **HydraLoRA** (Tian et al., 2024) shares asymmetric projections but keeps multiple trainable blocks.

MOELoRA (Liu et al., 2023) densely routes k trainable (A_i, B_i) pairs with input-dependent softmax weights. We use it as a *representative* router-based multi-LoRA benchmark—not sparse transformer MoE (no top-k FFN experts). Related expert-mixture PEFT methods differ in training: contrastive **MoELoRA** (Luo et al., 2024), post-hoc **MoLE** (Wu et al., 2024), and sparse/staged **MixLoRA** (Li et al., 2024)/**MoA** (Feng et al., 2024); we do not re-run every variant, but MOELoRA already anchors this family.

Table 1 positions ID-LoRA: like MOELoRA/HydraLoRA it routes over experts, but freezes clustered MID rows as $\{A_i\}$, trains one shared B, and applies RB to widen the *composed update dimension* $k{\cdot}r/s$ (we avoid the term "effective rank" for this concatenated width). In practice, this means ID-LoRA targets regimes where multiple adapter experts are useful but training k full LoRA blocks is too costly; the frozen pivot design trades flexibility in A_i for efficiency in B and routing.

Method	Trainable A_i	Trainable B	Routing	Trainable cost
LoRA	yes (1)	yes (1)	—	$\mathcal{O}(rd)$
LoRAHub	yes (k)	yes (k)	inference mix	$\mathcal{O}(krd)$
MoELoRA	yes (k)	yes (k)	dense softmax	$\mathcal{O}(krd)$
HydraLoRA	shared asym.	multiple blocks	task-specific	$\mathcal{O}(krd)$
ID-LoRA	frozen MID (k)	shared (1)	dense softmax	$\mathcal{O}(rd/s{+}r)$

Table 1: Positioning ID-LoRA among multi-LoRA and router-based PEFT methods (d hidden size, k experts, rank r, split s). MOELoRA (Liu et al., 2023) denotes dense router-based multi-LoRA, not sparse transformer MoE.

3 Method

Background. LoRA (Hu et al., 2022) confines adaptation to low-rank matrices $A \in \mathbb{R}^{r\times d}$ and $B \in \mathbb{R}^{d\times r}$ while freezing W:

$$u_t = Wh_t + \Delta W h_t = Wh_t + \frac{\alpha}{r} BAh_t \quad (1)$$

where h_t is the hidden state, α the scaling factor, and B is zero-initialized so $\Delta W{=}0$ at start. Matrix Interpolative Decomposition (MID) (Horn and Johnson, 2012) reuses pretrained rows as a skeleton: $\Delta W \approx BW_{[S,:]}$ with $|S|{=}r$, obtained by minimizing $\|BW_{[S,:]} - \Delta W\|_F$. Selecting informative pivot rows motivates our cluster-local PMRC rule.

Figure 1 summarizes the overall architecture and how PMRC, routing, and RB interact during the forward pass.

3.1 Parameter Matrix Row Clustering

Given a parameter matrix $W \in \mathbb{R}^{d\times d}$, we treat each row as a data point and partition all rows into k clusters via constrained K-means with minimum cluster size (Levy-Kramer, 2018) under the Euclidean metric. Within each cluster C_i ($i = 1, \ldots, k$), we select the r rows closest to centroid μ_i and stack them into a frozen basis $A_i \in \mathbb{R}^{r\times d}$. This yields $\{A_1, A_2, \ldots, A_k\}$, each capturing cluster-local row-space structure of W rather than a single global subspace. In our main experiments, $k{=}4$ and r is chosen from $\{32, 64, 128\}$ depending on the parameter budget (Section 4.2). PMRC runs once offline per module before training; cost and storage details are summarized in Section 4.2. We provide a comparative analysis justifying MID initialization against SVD and random initialization in Appendix B.6.

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.

$$u_t = Wh_t + \sum_{i=1}^{k} \alpha_i B(A_i h_t) \quad (2)$$

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 $\alpha_i = \mathrm{softmax}_i(\langle T, x_i\rangle)$, 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.11 reports router activation statistics across task types, revealing systematic differences that motivate learnable routing weights rather than fixed α values.

3.2 Rank Boosting

RB widens the *composed update* (concatenating k expert outputs after a shared B) for multi-task modeling while keeping trainable parameters nearly fixed by reusing a single shared projection. We partition each expert activation $x_i = A_i h_t \in \mathbb{R}^r$ and apply the shared B before concatenation (Figure 1 (b)):

$$x_i = A_i h_t,\ u_t = Wh_t + \sum_{i=1}^{k} \alpha_i f_c([Bx_i^1, Bx_i^2]) \quad (3)$$

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.

Capacity analysis. With $k{=}4$ and $s{=}2$, RB yields composed dimension 256 at nominal $r{=}128$ while keeping B at size $\mathbb{R}^{d/2\times r/2}$. Compared with MoELoRA (Liu et al., 2023), which learns k independent (A_i, B_i) pairs at $\mathcal{O}(krd)$ cost, ID-LoRA freezes clustered rows and learns one shared B plus router $T \in \mathbb{R}^r$ at $\mathcal{O}(rd/s{+}r)$; concatenation after RB widens the composed update dimension without training k separate B_i matrices.

3.3 Theoretical Analysis of Clustered Decomposition

We analyze PMRC through Matrix Interpolative Decomposition (CUR). **Assumption 1** (*clustering structure*) posits that related tasks' optimal updates occupy k low-rank subspaces; **Assumption 2** (*shared-basis composition*) posits that each

subspace is generated by one shared B and cluster-specific coefficients A_i. **Theorem 1** shows that, under these assumptions, cluster-aware reconstruction achieves a strictly tighter error bound than a single global low-rank adapter when inter-cluster discrepancy is non-negligible ($\Delta>0$). **Theorem 2** (pivot stability) shows that cluster-local pivots better preserve cluster-specific directions than global pivot sampling, motivating centroid-nearest rows within PMRC rather than unstructured global row choices.

Theory-to-module mapping. Assumption 1 $\rightarrow$ constrained k-means PMRC; Assumption 2 $\rightarrow$ one shared B over frozen $\{A_i\}$; Theorem 2 $\rightarrow$ PMRC vs. SS/RS pivot ablation (Table 10); Theorem 1 $\rightarrow$ multi-task gains vs. global LoRA/MOELoRA at matched budget (Tables 6, 4); learnable routing resolves *which* expert to use at each token but is not part of the reconstruction bound; RB widens the composed update dimension outside the theorems (Section 4.6.1). Full proofs appear in Appendix A.

4 Experiments

We evaluate ID-LoRA on single-task and multi-task benchmarks spanning math, code, knowledge, commonsense, and safety; report training/inference cost with guidance on k and r (Section 4.2); and ablate PMRC, RB, and pivot selection under matched trainable budgets. All main results use the default LLaMA-3-8B configuration (k=4, r=128, RB, $\sim$0.56% params) unless a table states otherwise; Mistral-7B and parameter-parity sweeps test backbone and budget generalization.

4.1 Experimental Setup

We compare ID-LoRA with FFT, LoRA (Hu et al., 2022), DoRA (Liu et al., 2024), and HydraLoRA (Tian et al., 2024); multi-task runs additionally include MOELoRA (Liu et al., 2023) (Liu et al., 2023) and VeRA (Kopiczko et al., 2024). Appendix B.8 and B.10 report LoRA-XS (Bałazy et al., 2024) and LoRI-D (Zhang et al., 2025). All baseline and ID-LoRA runs share a unified implementation on the Hugging Face `peft` library (Mangrulkar et al., 2023): built-in methods use the corresponding adapter types, and ID-LoRA is registered as a custom `PeftModel` so adapter injection, optimization, and checkpointing follow the same interface across methods.

Tasks. Single-task: GSM8K (Math) (Cobbe et al., 2021), HumanEval (Code) (Chen et al., 2021), and HEx-PHI (Safety) (Qi et al., 2023) after fine-tuning on GSM8K, CodeAlpaca (Chaudhary, 2023), and Saferpaca (Bianchi et al., 2023). Multi-task: joint fine-tuning on Alpaca (Taori et al., 2023), GSM8K train, and 50% CodeAlpaca; evaluation on GSM8K, HumanEval, MMLU (Hendrycks et al., 2021), and CommonsenseQA (Talmor et al., 2019).

LoRA	DoRA	HydraLoRA	MoELoRA	ID-LoRA
1.03	1.81	1.48	1.99	1.27

Table 2: Multi-task training wall-clock on LLaMA-3-8B (single A800 GPU, b_s=32, `train_step`=1000). Time unit is hour.

Hardware and seeds. Main experiments use two NVIDIA A800 GPUs with LLaMA-3-8B (Grattafiori et al., 2024) and Mistral-7B (Jiang et al., 2023); LoRA, DoRA, and HydraLoRA share ID-LoRA hyperparameters except where rank or adapter count must differ for budget matching. Multi-task and parameter-parity results report mean$\pm$std over seeds $\{42, 123, 456\}$; single-task headline numbers are single-seed with Wilcoxon tests where marked. Hyperparameters are in Appendix B.2.

4.2 Practical Cost and Usage Guidance

Default configuration. Our main LLaMA-3-8B experiments apply ID-LoRA to seven weight modules (`q,k,v,o,gate,up,down_proj`) with k=4, nominal r=128, split factor s=2, and RB enabled ($\sim$0.56% trainable parameters). Mistral-7B uses the same k and RB settings at $\sim$0.62% trainable parameters.

Preprocessing (PMRC). PMRC is applied offline once per targeted module before any fine-tuning. On LLaMA-3-8B (d=4096), clustering all seven modules takes $\sim$12 minutes on one NVIDIA A800 GPU (dominated by constrained k-means on $\sim$28K rows per matrix). Cost scales roughly linearly with the number of adapted modules and is amortized across subsequent training runs; at inference, only cluster indices, frozen row lookups, shared B, and the router need to be retained.

Training wall-clock. Table 2 reports multi-task wall-clock on one A800 GPU (b_s=32, `train_step`=1000): ID-LoRA needs 1.27 h vs. 1.03 h for LoRA ($\sim$1.23$\times$), below DoRA/HydraLoRA/MoELoRA. Main results (Tables 6–5) use two A800 GPUs; the single-GPU study isolates per-method training efficiency.

Inference overhead. Figure 2 reports latency and extra GPU memory on LLaMA-3-8B multi-

k	MATH	MMLU	CQA	CODE P@1	CODE P@5	CODE P@10
2	56.1	48.9	48.2	39.5	52.0	56.9
4	**59.6**	**51.0**	47.4	41.7	**54.5**	**58.5**
6	59.6	49.7	47.8	**42.9**	52.7	57.2
8	58.9	50.4	46.2	41.6	53.2	57.6

Table 3: Cluster-number ablation on LLaMA-3-8B multi-task data (PMRC+RB, r=128, s=2). Bold: default k=4.

task checkpoints: ID-LoRA adds $\sim$0.5% latency vs. LoRA while using $\sim$45% less extra GPU memory than MoELoRA/HydraLoRA; merged LoRA/DoRA paths add negligible overhead.

Choosing k. We default to k=4. Table 3 sweeps $k \in \{2, 4, 6, 8\}$ at fixed r=128 with RB: k=4 yields the best MMLU/code balance, while k=6 slightly favors code Pass@1 at a small MMLU drop. Very small k=2 underfits heterogeneous skills; larger k increases routing overhead without consistent gains.

Choosing r. Table 4 compares LoRA and ID-LoRA at matched trainable budgets—the setting closest to Theorem 1's "same parameter envelope, different factorization" claim. ID-LoRA wins MMLU/code at the default tier (0.56% vs. LoRA 0.52%) while staying within 0.4 points on math; at the high tier it nearly matches LoRA on math (60.7 vs. 60.8) but still leads on MMLU/code. Use r=128 **with RB** by default; raise toward r=256 only if knowledge/code metrics still improve. CQA can peak then drop at higher capacity (Section 4.5).

Figure 4 visualizes the parameter-parity trends summarized in Table 4: ID-LoRA's advantage on MMLU and code widens at the default and high budget tiers while math stays within a narrow band. This pattern matches Theorem 1's prediction that a *different factorization* under the same budget envelope can help heterogeneous tasks without uniformly improving every skill.

4.3 Experiments on Single-Task

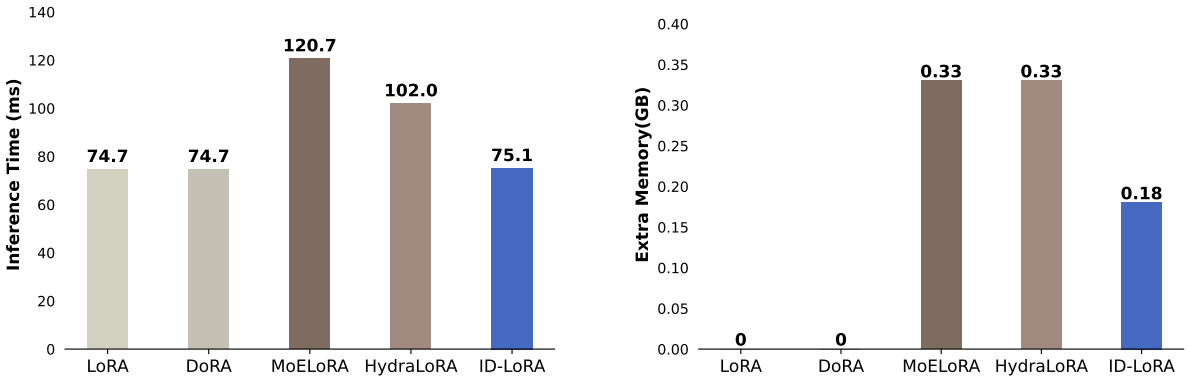
Table 5 evaluates whether ID-LoRA's clustered MID design helps when only one task distribution is present—a regime where Assumption 1's inter-cluster gap Δ is small and theory predicts smaller advantage over a global LoRA factor. ID-LoRA uses 0.56% (LLaMA-3) or 0.62% (Mistral) trainable parameters yet remains competitive on most columns; it is *not* uniformly best. On GSM8K, ID-LoRA ties the best LLaMA-3 score (63.5%) at $\sim$50% fewer parameters; on Mistral math it trails HydraLoRA (58.1 vs. 58.3) and LoRA on Pass@1 (33.5 vs. 33.8). The largest single-task margins appear on HumanEval Pass@10 (LLaMA-3: 55.0 vs. 50.8 LoRA) and HEx-PHI on both backbones—tasks where structured, pivot-based updates appear more useful than unconstrained rank expansion. This pattern foreshadows multi-task results: ID-LoRA is strongest when heterogeneous skills must coexist under one adapter.

4.4 Experiments on Multi-Task

Table 6 is the primary test of our multi-task claims under mixed Alpaca/GSM8K/CodeAlpaca training (three seeds on LLaMA-3-8B). **Claim 1 (parameter efficiency).** ID-LoRA uses $\sim$46% fewer trainable weights than LoRA (0.56% vs. 1.03%) yet leads on five of six LLaMA-3 columns. Turning to heterogeneous subspaces, **Claim 2** shows gains concentrate on knowledge and code—MMLU (+2.9), code Pass@10 (+6.6)—where Theorem 1 predicts benefit from cluster-local factors when task updates diverge; math trails LoRA by 1.2 points (59.6$\pm$0.5 vs. 60.8$\pm$0.7), suggesting numeric reasoning still favors a single unconstrained low-rank path. Finally, **Claim 3 (vs. router multi-LoRA)**: MOELoRA underperforms despite similar routing capacity (e.g., MMLU 41.5, code Pass@10 47.1), supporting the hypothesis that freezing clustered MID rows plus one shared B is more parameter-efficient than learning k full (A_i, B_i) pairs. VeRA is extremely small (0.0178%) but does not close the code gap. On Mistral, ID-LoRA leads MMLU/CQA and ties top math/code Pass@1 within $\leq$0.4 points (Section 4.5).

4.5 Result Stability

Table 7 checks whether the multi-task conclusions above are seed artifacts. Variances stay below 1.1 points on MMLU, CQA, and code; the MMLU advantage (+2.9) and code Pass@1 gain (+2.7) persist across seeds, while the math deficit (-1.2) is also stable—so the theory-motivated *split* between knowledge/code gains and math trade-offs is reproducible, not a single lucky run. Parameter-parity trends in Table 4 are similarly stable for MATH/MMLU/code as budget grows; CQA can peak then drop at higher capacity (Appendix Table 21), a capacity effect orthogonal to PMRC/RB design. Together with the component ablations (Table 8), these statistics support reporting LLaMA-3 multi-task numbers as mean$\pm$std while treating single-task and Mistral columns as indicative rather than variance-adjusted.



(a) Inference time (b) Extra GPU memory

Figure 2: Inference cost on LLaMA-3-8B multi-task checkpoints (A800 GPU).

Budget (%)	MATH	MMLU	CQA	Code P@10	Setting
L:0.32 / I:0.32	57.9 / 60.0	47.2 / 51.0	47.3 / 61.0	34.5 / 53.7	mid
L:0.52 / I:**0.56**	59.4 / **59.6**	47.5 / **51.0**	37.8 / 47.4	42.2 / **58.5**	default
L:1.03 / I:0.99	**60.8** / 60.7	48.1 / **51.8**	46.9 / 64.8	51.9 / **59.9**	high

Table 4: Parameter-parity on LLaMA-3-8B multi-task data. Each row compares LoRA (L) and ID-LoRA (I) at matched trainable-parameter budgets; best per column bolded. Full sweep in Appendix Table 21.

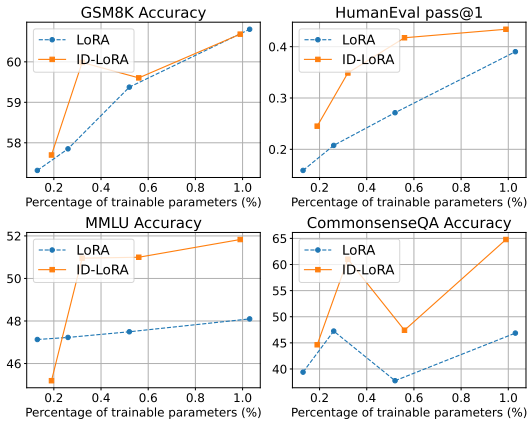


Figure 3: Parameter-parity comparison between ID-LoRA and LoRA on LLaMA-3-8B multi-task benchmarks. Full numeric sweep in Appendix Table 21.

4.6 Ablation Analysis

We ablate each module against the theoretical claims in Section 3.3, holding trainable budget fixed where possible. Table 8 first isolates PMRC/MID (LoRA vs. ID-LoRA w/o RB) and RB (w/o RB vs. PMRC+RB); Appendix Table 22 tests whether pretrained-row pivots beat SVD/random init (supporting Assumption 2's use of W structure). Tables 9 and 10 below target RB capacity and Theorem 2 pivot quality, respectively.

Comparing LoRA with ID-LoRA (w/o RB), replacing a learned A with PMRC-selected frozen rows plus shared B improves MMLU (+2.1) and code Pass@10 (+1.8) at $\sim$42% fewer trainable parameters (0.60% vs. 1.03%), supporting Assumption 2's reuse of pretrained structure. Enabling RB at near-matched budget ($\sim$0.56%) further lifts code Pass@10 to 58.5 while keeping math comparable (59.6 vs. 59.4), confirming that widening the composed update—not trainable rank alone—drives heterogeneous multi-task gains.

4.6.1 Rank Boosting under Matched Controls

This ablation tests whether widening the *composed update* (not trainable rank alone) explains multi-task gains. With PMRC and routing fixed (k=4, s=2), w/o RB is capped at r=32 (composed dim. 32) at $\sim$0.60% params, whereas PMRC+RB reaches r=128 (composed dim. 256) at $\sim$0.56%. PMRC+RB improves MMLU (+0.8), CQA (+1.0), and code Pass@10 (+4.8) with comparable math (59.6 vs. 59.4), supporting the claim that RB supplies task-heterogeneity capacity beyond Theorem 1's factorization. Critically, at the same nominal r=128, scaling B without RB still trails on code (56.1 vs. 58.5 Pass@10 at 0.99% params; Appendix Table 17)—widening via concatenation matters, not parameter count alone.

4.6.2 PMRC Pivot Selection

Theorem 2 predicts that cluster-local pivots outperform *global* pivot rules when task clusters occupy different row subspaces. We therefore compare PMRC with Serial Selection (SS; contiguous rows, mimicking poor global coverage) and Random Selection (RS; uniform global sampling), fixing RB, k, r, s, and budget ($\sim$0.56%). Table 10 shows

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 5: 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±0.5	**51.0**±0.6	**47.4**±0.8	**41.7**±0.9	**54.5**±1.0	**58.5**±1.1
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 6: Performance comparison on a multi-task suite (Math/GSM8K, MMLU, CommonsenseQA, HumanEval). LLaMA-3 ID-LoRA entries are mean±std over three seeds; other cells are single-run. **Bold**: best in column. "CQA": CommonsenseQA. †: Wilcoxon vs. LoRA (p<0.05).

Metric	LoRA (r=32)	ID-LoRA	Δ
MATH	60.8±0.7	59.6±0.5	−1.2
MMLU	48.1±0.9	51.0±0.6	+2.9
CQA	46.9±1.0	47.4±0.8	+0.5
Code Pass@1	39.0±1.1	41.7±0.9	+2.7

Table 7: Mean±std over three seeds (LLaMA-3-8B multi-task).

Method	MATH	MMLU	CQA	CODE		
				P@1	P@5	P@10
LoRA	60.8	48.1	46.9	39.0	48.8	51.9
ID-LoRA (w/o RB)	59.4	50.2	46.4	40.2	50.8	53.7
ID-LoRA (All)	59.6	51.0	47.4	41.7	54.5	58.5

Table 8: Core component ablation on LLaMA-3-8B multi-task data. LoRA vs. ID-LoRA (w/o RB) isolates PMRC/MID; w/o RB vs. All(PMRC+RB) isolates RB under comparable trainable budgets.

PMRC improves code Pass@10 by +2.3/+3.1 over SS/RS and matches or exceeds them on MMLU, with comparable math—the largest separation appears on code, the most heterogeneous skill in the mix. This is the direct empirical counterpart to Theorem 2; together with Appendix Table 22, it supports using pretrained *cluster-local* MID rows rather than SVD/global random rows as $\{A_i\}$.

Variant	Params	r	MMLU	CQA	Code P@10
w/o RB	0.60	32	50.2	46.4	53.7
PMRC+RB	0.56	128	51.0	47.4	58.5

Table 9: Budget-matched RB ablation (LLaMA-3-8B). Rank-matched controls in Appendix Table 17.

Pivot rule	MATH	MMLU	CQA	Code P@10
SS+RB	58.8	50.7	47.7	56.2
RS+RB	59.2	51.0	47.2	55.4
PMRC+RB	59.6	51.0	47.4	58.5

Table 10: PMRC pivot-selection ablation on LLaMA-3-8B multi-task data. All variants share the same trainable-parameter budget (∼0.56% of backbone weights) and RB configuration (r=128, s=2, k=4).

4.7 Discussion

Taken together, the single-task, multi-task, and ablation results suggest a consistent story rather than isolated wins. ID-LoRA's advantage is largest when *multiple* skill types must coexist in one adapter: cluster-local MID pivots and RB supply capacity for knowledge and code subspaces that a

single global LoRA factor under-utilizes at comparable trainable cost (Tables 6, 8, 9). By contrast, when only one dominant task distribution is present, the inter-cluster gap predicted by Assumption 1 shrinks and ID-LoRA behaves like a competitive but not uniformly dominant adapter (Table 5). Math is the recurring exception: both multi-task GSM8K (-1.2 vs. LoRA; Table 7) and some Mistral single-task columns favor unconstrained low-rank updates, implying that numeric reasoning may require less structured pivot reuse.

Router-based baselines clarify *why* parameter efficiency matters beyond routing alone. MOELoRA shares input-dependent mixing yet trains k independent (A_i, B_i) pairs; its lag on MMLU/code at similar trainable scale supports the design choice to freeze clustered rows and learn only shared B plus a lightweight router (Table 1). PMRC pivot quality (Table 10) and initialization comparisons (Appendix Table 22) further indicate that *where* pivots come from—cluster centroids of W, not global SVD or random rows—is as important as having multiple experts. Inference cost (Figure 2) remains near LoRA despite routing, so the practical bottleneck is training-time adapter memory rather than deployment latency. These patterns motivate the usage guidance below and the scoped claims in Limitations.

4.8 Summary of Findings

We summarize how the claims in Section 3.3 map to experiments. In short: ID-LoRA is a strong *multi-task, mid-budget* PEFT choice when knowledge and code matter most; it does not uniformly dominate math or every backbone. Router statistics (Appendix B.11) show task-dependent expert usage, consistent with input-adaptive routing beyond fixed mixing.

When to use ID-LoRA. For LLMs (LLaMA-3-8B and Mistral-7B) under multi-task instruction tuning with mixed knowledge, commonsense, and code objectives, ID-LoRA at $\sim 0.56\%$ trainable parameters on LLaMA-3-8B ($\sim 0.62\%$ on Mistral-7B) is most compelling when MMLU/code gains outweigh a modest math trade-off (Tables 6, 4, 7). Single-task math-only fine-tuning or settings requiring a single global unconstrained LoRA path may still favor standard LoRA (Table 5). The PMRC/RB design aligns with Theorems 1–2 when tasks occupy distinct row subspaces; input-dependent routing then selects among frozen cluster-local pivots rather than fixed mixing coefficients.

Claim	Main evidence
Clustered factorization helps heterogeneous multi-task	MMLU/code gains at 0.56% params (Tab. 6); MOELoRA lags with similar routing
Shared B + frozen $\{A_i\}$ beats k trainable LoRAs	Tab. 1, 8; $\sim 46\%$ fewer params than LoRA
Cluster-local pivots $>$ global rows (Thm. 2)	Tab. 10 vs. SS/RS; Tab. 22 vs. SVD/random
RB widens composed update, not just params	Tab. 9, 17; code +4.8 at matched budget
Math not uniformly improved	-1.2 GSM8K vs. LoRA (Tab. 7); single-task math ties/trails on Mistral

Table 11: Mapping from theoretical/design claims to main experimental evidence.

Table 11 makes explicit which results support each design choice: multi-task MMLU/code tables and MOELoRA comparisons anchor Theorem 1; PMRC/SS/RS and initialization ablations anchor Theorem 2; RB and rank-matched controls isolate composed-update widening; and the stable math deficit in Table 7 bounds our claims. None of the rows alone implies universal dominance—rather, they specify *when* the clustered MID factorization is expected to help.

5 Conclusion

We propose ID-LoRA, a PEFT framework that freezes clustered pretrained rows as $\{A_i\}$, trains one shared B, routes over experts, and applies RB to widen the composed update dimension at $\mathcal{O}(rd/s+r)$ trainable cost—far below router-based multi-LoRA baselines that learn k full (A_i, B_i) pairs (Table 1). Cluster-aware CUR-style analysis (Section 3.3) motivates PMRC and pivot selection; component ablations confirm that frozen MID rows, shared composition, and RB each contribute measurable gains under matched budgets. On LLaMA-3-8B and Mistral-7B, ID-LoRA improves several knowledge and code metrics at $\sim 46\%$ fewer parameters than LoRA, with competitive single-task safety/code results, but multi-task math is often weaker. Future work includes dynamic cluster count, broader backbone coverage, and systematic comparison with MoLE, MixLoRA, and MoA under unified multi-task protocols.

6 Limitations

We scope claims to the tasks, backbones, and budgets evaluated above. ID-LoRA targets multi-task, mid-budget instruction tuning on LLaMA-3-8B and Mistral-7B with mixed knowledge, commonsense, and code objectives when knowledge and code gains justify a modest math trade-off (Sections 4.8 and 4.7); it should not be read as a universal LoRA replacement.

Assumptions behind the design. PMRC selects pivots from rows of frozen pretrained weights. Targets whose optimal updates are poorly aligned with the row-space clusters of W—including highly out-of-distribution tasks—may see limited benefit. Theorems 1–2 formalize clustered MID reconstruction under Assumptions 1–2 but do not bound input-dependent routing or RB; their role is supported only by ablations (Section 4.6).

Task- and backbone-dependent gains. On LLaMA-3-8B multi-task data, ID-LoRA leads on MMLU and code at $\sim$46% fewer trainable parameters than LoRA, yet GSM8K trails by a stable 1.2 points over three seeds (Table 7). Single-task math is competitive but not uniformly best (Table 5). On Mistral-7B, several columns tie or trail the strongest baseline within $\leq$0.4 points (Table 6). We do not claim to uniformly break the rank–parameter trade-off.

Coverage of experiments and baselines. Variance-aware reporting is limited to LLaMA-3-8B multi-task runs; Mistral multi-task entries, most baseline cells, and single-task headline numbers are single-run. MOELoRA serves as a representative router-based multi-LoRA baseline, but we do not re-run every related method (MoLE, MixLoRA, MoA). Results on Qwen3-8B, VeRA, LoRA-XS, and LoRI-D appear only in the appendix (Sections B.8, B.9, and B.10).

Cost, tuning, and ablation caveats. PMRC incurs a one-time offline clustering cost per adapted module (Section 4.2), and multi-task training is $\sim$1.23$\times$ the wall-clock of LoRA on one A800 GPU. Performance depends on cluster count k (Table 3) and can be non-monotonic in capacity on CQA (Appendix Table 21). RB gains are interpreted alongside budget-matched controls (Table 9); showing that widening the composed update at fixed nominal r=128 requires a larger trainable envelope (0.99% vs. 0.56%; Appendix Table 17).

References

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.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. 2024. Mixture-of-loras: An efficient multitask tuning for large language models.

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*.

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*.

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.

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

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.

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.

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

Sourab Mangrulkar, Sylvain Gervais, Robert Kirk, Thomas Wolf, Younes Belkada, Edward Beeching, Nathan Cooper, Lewis Tunstall, David von Crämer, Fangxu Liu, Shengyi Huang, Sayak Paul, and Benjamin Bossan. 2023. PEFT: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

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.

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. 2023. 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.

Xun Wu, Shaohan Huang, and Furu Wei. 2024. Mixture of lora experts. *arXiv preprint arXiv:2404.13628*.

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 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:*

$$\forall i, j \in \mathcal{C}_l, \quad \textit{Rank}(W_i^* - W_j^*) \leq r_l < \min(d_1, d_2) \tag{4}$$

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:*

$$W_i^* = BA_i + E_i, \quad \|E_i\|_F^2 \leq \epsilon \tag{5}$$

where $A_i \in \mathbb{R}^{r_g \times d_1}$ is a task-specific matrix 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,*

$$\max_{\textit{bad pivots}} \|W - \tilde{W}_{\text{CUR}}\|_F^2 > \min_{\textit{good pivots}} \|W - \tilde{W}_{\text{CUR}}\|_F^2 \tag{6}$$

A.2 Core Definitions

Definition 2 (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:*

$$d(i, j) = \|W_i^* - W_j^*\|_F^2 \tag{7}$$

Definition 3 (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

$$\min_B \sum_{i=1}^{m} \|W_i^* - BA_{l(i)}\|_F^2, \tag{8}$$

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

Definition 4 (Multi-Task Reconstruction) *The parameter matrix for task i is reconstructed as*

$$\tilde{W}_i = BA_{l(i)} \tag{9}$$

Definition 5 (CUR Decomposition) *Given a matrix $W \in \mathbb{R}^{d\times d}$, a* CUR *decomposition approximates W as a product of three matrices:*

$$W \approx CUR \tag{10}$$

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

$$\sum_{i=1}^{m} \|W_i^* - \tilde{W}_i\|_F^2 \le \sum_{i=1}^{m} \|W_i^* - \tilde{W}_i^{global}\|_F^2 - \Delta \tag{11}$$

where $\Delta = \sum_{l=1}^{k}\sum_{i\in\mathcal{C}_l} \|B(A_{l(i)} - A^{global})\|_F^2 \ge 0$. 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

$$\left\|W_i^* - \tilde{W}_i^{\text{global}}\right\|_F^2 = \left\|B(A_i - A^{\text{global}}) + E_i\right\|_F^2. \tag{12}$$

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

$$\left\|W_i^* - \tilde{W}_i\right\|_F^2 = \left\|B(A_i - A_{l(i)}) + E_i\right\|_F^2. \tag{13}$$

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

$$\|A_i - A_{l(i)}\|_F^2 \le \|A_i - A^{\text{global}}\|_F^2, \tag{14}$$

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

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:*

$$\mathbb{E}_{\mathcal{P}}\left[\max_{1\le i\le m} \|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F^2\right] \ge \mathbb{E}_{\mathcal{P}_l}\left[\max_{1\le i\le m} \|W_i^* - \tilde{W}_i\|_F^2\right], \tag{15}$$

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:

$$\tilde{W}_i^{\text{CUR}} = C_{\mathcal{P}} U_{\mathcal{P}} R_{\mathcal{P}}, \quad \mathcal{P} \subseteq [d_1]\times[d_2], |\mathcal{P}| = s \tag{16}$$

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

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

$$\tilde{W}_i = C_{\mathcal{P}_l} U_{\mathcal{P}_l} R_{\mathcal{P}_l}, \mathcal{P}_l \sim \text{cluster-specific distribution} \tag{17}$$

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:

$$\mathbb{P}_{\mathcal{P}}(\mathbf{v}_{l^*} \notin \text{span}(C_{\mathcal{P}})) \ge \eta_{l^*} > 0 \tag{18}$$

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:

$$\|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F \ge \gamma_{l^*} \quad \forall i \in \mathcal{C}_{l^*} \tag{19}$$

Thus:

$$\mathbb{E}_{\mathcal{P}}\left[\max_i \|W_i^* - \tilde{W}_i^{\text{CUR}}\|_F\right] \ge \gamma_{l^*}\cdot\eta_{l^*} \tag{20}$$

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:

$$\mathbb{P}_{\mathcal{P}_l}(\mathbf{v}_l \in \text{span}(C_{\mathcal{P}_l})) \geq 1 - \epsilon_l \quad (\epsilon_l \leq e^{-c|\mathcal{P}_l|}) \tag{21}$$

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

(2) Error Decomposition:

$$\|W_i^* - \tilde{W}_i\|_F \leq \underbrace{\|M_l - \tilde{M}_l\|_F}_{\text{cluster center error}} + \underbrace{\|W_i^* - M_l\|_F}_{\leq \delta_l} \tag{22}$$

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:

$$\|M_l - \tilde{M}_l\|_F \leq (1 + \sqrt{r_l})\|M_l - M_l^{(r_l)}\|_F = 0 \tag{23}$$

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

$$\mathcal{E}_l^{\text{local}} \leq \delta_l \quad \text{with probability} \geq 1 - \epsilon_l \tag{24}$$

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

(4) Expectation Control:

$$\mathbb{E}[\mathcal{E}_l^{\text{local}}] \leq \delta_l(1-\epsilon_l) + (\text{diam}(\mathcal{C}_l) + \|M_l\|_F)\epsilon_l \leq \delta_l + \mathcal{O}(\epsilon_l) \tag{25}$$

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:

$$\mathbb{E}[\mathcal{E}_{\text{global}}] \geq \gamma_{l^*}\eta_{l^*}, \quad \mathbb{E}[\mathcal{E}_{\text{local}}] \leq \max_l(\delta_l + \mathcal{O}(\epsilon_l)) \tag{26}$$

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 experimental details and supplementary results omitted from the main text. Hyperparameter tables, theoretical proofs, and extended ablations appear below; preprocessing/training/inference costs and k/r selection guidance are reported in Section 4.2.

B.1 Positioning among Multi-LoRA Methods

Table 1 in the main text summarizes trainable cost and routing among nearby multi-LoRA designs.

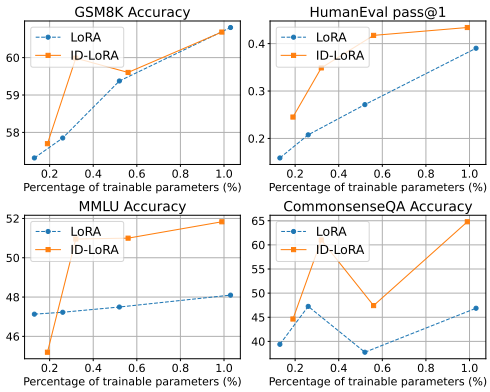


Figure 4: Parameter-parity comparison between ID-LoRA and LoRA on LLaMA-3-8B multi-task benchmarks.

Table 14 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.2 Hyperparameter Settings

In this section, we provide detailed hyperparameter settings for the main experiments in the paper, which are presented in Tables 12 and 13. Table 12 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 13 shows the hyperparameter settings for multi-task scenarios.

B.3 Cost, Parameter Parity, and Stability

Figure 4 visualizes parameter-parity trends; the full numeric sweep is in Table 21.

Extended seed-variance analysis and CQA non-monotonicity at high rank appear in Section 4.5 and Table 21.

B.4 Ablation Study Design

All component ablations are conducted under comparable trainable-parameter budgets. Rank Boosting (RB) is designed so that, at a fixed budget, ID-LoRA can adopt a higher nominal rank and a wider composed update dimension than a standard LoRA adapter of similar trainable cost—this is an intended advantage of RB, not an uncontrolled confound, when interpreted alongside the matched-budget and matched-rank controls in Table 9 (main text).

The core ablation logic isolates two contributions (Table 8 in the main text):

- **PMRC / MID (frozen clustered bases).**

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 12: 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.

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

LoRA vs. ID-LoRA (w/o RB) replaces a learned down-projection A with frozen rows selected by PMRC while keeping trainable cost in a comparable regime. ID-LoRA (w/o RB) improves MMLU and code over LoRA despite using $\sim$42% fewer trainable parameters (0.60% vs. 1.03%), supporting the benefit of clustered pretrained pivots and shared B.

- **RB (composed-update widening).** ID-LoRA (w/o RB) vs. PMRC+RB at near-matched trainable budgets ($\sim$0.60% vs. 0.56%). PMRC+RB reaches nominal r=128 and composed dimension 256, whereas w/o RB is limited to r=32 at the same budget tier; gains on MMLU and code confirm RB's role in widening the composed update without proportional parameter growth.

Pivot-selection ablation is in main text Table 10; initialization results are in Table 22 below.

B.5 Ablation: Impact of Pretrained Parameter Cluster Number k

Cluster-count sensitivity for LLaMA-3-8B is reported in main text Table 3; a complementary sweep on LLaMA-3.2-3B is in Table 15.

B.6 Ablation: Impact of Initialization Strategies

We evaluated Matrix Interpolative Decomposition (MID) against SVD and random row selection while keeping RB and budget fixed. Table 22 shows MID consistently outperforms both; SVD lags substantially on MMLU (35.2 vs. 51.0), likely because global principal components miss cluster-local structure needed for heterogeneous multi-task adaptation.

B.7 Ablation: Interpreting RB under Parameter-Parity Budgets

Table 9 in the main text reports RB ablations under matched trainable-budget and matched nominal-rank controls; Table 16 summarizes the resulting budget/composed-dimension trade-offs, and Table 17 provides three-seed statistics for the rank-matched block. The rank-matched control shows that scaling B to r=128 without RB (0.99% params) still trails PMRC+RB on code Pass@10 (56.1 vs. 58.5), supporting RB-induced widening of the composed update dimension rather than trainable capacity alone. We do not use "effective rank" for this concatenated width.

B.8 Additional Baselines: LoRA-XS and VeRA under Multi-Task Instruction-Tuning

Table 18 shows that ID-LoRA exceeds LoRA-XS and VeRA on the reported LLaMA-3-8B multi-task benchmarks at a higher but still practical parameter budget. This supports reuse of pretrained structure via MID on knowledge/code tasks; it should not be read as uniform dominance over all PEFT methods or tasks (cf. math/Mistral results in the main table).

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

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 15: Cluster-number ablation on LLaMA-3.2-3B multi-task data (PMRC+RB, r=128). "COM": CommonsenseQA.

Method	**Params (%)**	r	**Composed dim.**
ID-LoRA (PMRC+RB)	0.56	128	$k{\cdot}r/s$
ID-LoRA (w/o RB)	0.60	32	r
ID-LoRA (w/o RB)	0.99	128	r

Table 16: Nominal rank, trainable budget, and composed update dimension for RB ablation (k=4, s=2). "Composed dim." denotes concatenated expert output width, not numerical rank or "effective rank."

Variant	**Params**	r	**MMLU**	**Code P@10**
w/o RB	0.60%	32	50.2±0.7	53.7±1.2
PMRC+RB	0.56%	128	51.0±0.6	58.5±1.1
w/o RB	0.99%	128	50.4±0.8	56.1±1.0

Table 17: Matched-rank control: same r=128 with vs. without RB at higher trainable budget (row 3). Near-budget comparison in rows 1–2.

B.9 Additional Results on a Newly Released Backbone: Qwen3-8B

Table 19 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 widening the composed update dimension alone provides, suggesting that incorporating stronger directionally constrained updates is a promising future direction.

B.10 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-LoRAs 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 20, ID-LoRA outperforms LoRI-D on all listed Qwen3-8B tasks in that additional study (math still slightly below LoRA in Table 19). We attribute gains on non-math tasks to ID-LoRAs trainable router, which learns input-dependent expert composition, whereas LoRI-D relies on a linear combination governed by manually specified hyperparameters, limiting its ability to adaptively resolve cross-task interference.

B.11 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 23.

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

Table 23 shows that Router 3 has the highest average activation overall, but the by-task statis-

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 18: 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 19: 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 20: Comparison with LoRI-D on Qwen3-8B under the same mixed-data multi-task training setup.

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

Table 21: Full parameter-parity comparison between LoRA and ID-LoRA (LLaMA-3-8B, multi-task). Main text Table 4 reports three representative budget tiers.

Init.	MATH	MMLU	CQA	Code P@10	Params (%)
Random	60.2	49.2	44.9	55.2	0.56
SVD	47.9	35.2	44.6	45.3	0.56
MID (Ours)	59.6	51.0	47.4	58.5	0.56

Table 22: Initialization ablation on LLaMA-3-8B multi-task data (PMRC+RB, r=128).