CD-LoRA: Consistency-Driven Low-Rank Adaptation for Multi-Task Fine-Tuning

Anonymous ACL submission

Abstract

While Multi-Task Learning (MTL) is essential for adapting Large Language Models (LLMs) to diverse domains, prevailing LoRA-based methods often rely on complex routing mechanisms to partition task-specific knowledge. We show that such routing-based designs suffer from an **adapter-level training–inference discrepancy** caused by stochastic routing decisions, and further characterize the residual **output-level training–inference discrepancy** after de-routing via a second-order Taylor analysis. Motivated by this two-level analysis, we propose **Consistency-Driven Low-Rank Adaptation (CD-LoRA)**. CD-LoRA eliminates routers to remove adapter-level routing discrepancy and introduces consistency-driven alignment to mitigate residual output-level mismatch by enforcing **representation congruence** in a shared low-rank space. Extensive experiments show that CD-LoRA consistently outperforms routing-based and multi-adapter baselines, offering a simpler, router-free, and more stable solution for multi-task PEFT. The code is available at the anonymous link CD-LoRA.

1 Introduction

LLMs have demonstrated remarkable performance across a wide range of Natural Language Processing (NLP) tasks (Brown et al., 2020). Despite their strong zero-shot generalization, adapting LLMs to specialized domains or evolving tasks remains essential (Lee et al., 2020). While Supervised Fine-Tuning (SFT) is the standard adaptation paradigm, Full Parameter Fine-Tuning (FPT) incurs prohibitive computational and memory costs as model scales grow (Lialin et al., 2023). This has motivated extensive research on Parameter-Efficient Fine-Tuning (PEFT)(Hu et al., 2022; Li and Liang, 2021), among which Low-Rank Adaptation (LoRA)(Hu et al., 2022) has emerged as a dominant approach.

Recent efforts (Liu et al., 2025) extend LoRA to Multi-Task Learning (MTL) by introducing task-specific or multi-head adapter structures. Early multi-adapter methods allocate independent low-rank parameters per task (Wang et al., 2024), while subsequent designs improve parameter efficiency by sharing components across tasks, such as a common A with multiple B_is (Tian et al., 2024). To further enhance flexibility, many approaches incorporate routing mechanisms inspired by Mixture-of-Experts (MoE)[1] (Shazeer et al., 2017), dynamically weighting adapter outputs and encouraging head diversity (Li et al., 2024). Figure 1 summarizes the architectural evolution of representative fine-tuning paradigms.

Collectively, these methods assume that effective multi-task adaptation relies on **dynamic routing**(as shown in subfigure (d) of Figure 1.) to disentangle **task-specific knowledge**.

However, our work challenges this paradigm. We analytically identify an *adapter-level training–inference discrepancy* (Section 3.1) induced by nonlinear routing, showing that input-dependent routing weights do not generally commute with stochastic dropout perturbations. We further analyze the residual *output-level discrepancy* after de-routing, where stochastic low-rank representations may still be amplified by nonlinear downstream Transformer layers. Beyond theoretical analysis, latent space visualizations (Section 3.2) suggest that maintaining coherent representations in the shared low-rank space is important for stable multi-task adaptation.

Motivated by this two-level analysis, we propose **Consistency-Driven Low-Rank Adaptation (CD-LoRA)**. CD-LoRA removes the adapter-level discrepancy by eliminating nonlinear routers, and further mitigates residual output-level mismatch through consistency-driven alignment in the shared

[1] In LoRA, MoE-style routing is used to dynamically compose lightweight LoRA adapters rather than to introduce full expert feed-forward networks into the backbone.

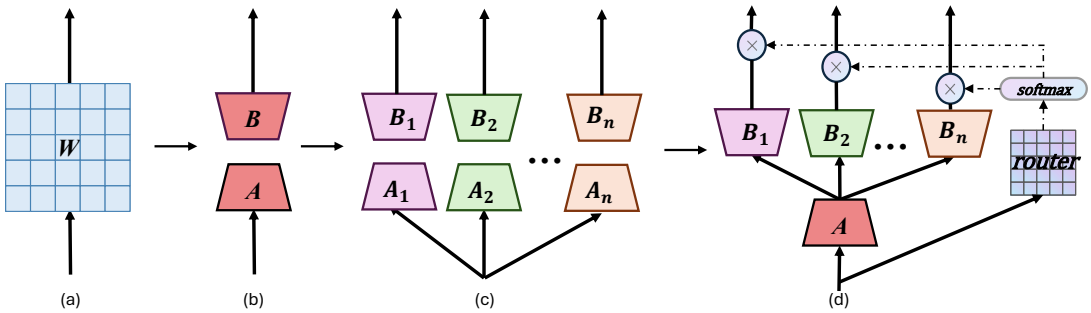

Figure 1: Architectural comparison of fine-tuning paradigms. (a) Full Parameter Fine-Tuning (FPT); (b) Vanilla LoRA; (c) Multi-Adapter architectures with task-specific low-rank modules; (d) Multi-Head or asymmetric LoRA variants with routing mechanisms.

low-rank space. Extensive experiments show that CD-LoRA not only consistently outperforms multi-adapter baselines but also moderately reduces computational overhead. Extensive experiments show that CD-LoRA not only consistently outperforms multi-adapter baselines but also moderately reduces computational overhead.

Our contributions are summarized as follows:

- We provide a two-level analysis of training–inference discrepancy, separating routing-induced adapter-level mismatch from residual output-level mismatch.
- We show that de-routing eliminates the adapter-level discrepancy and offers a simpler, more efficient alternative to dynamic routing.
- We propose consistency-driven alignment in the shared low-rank space to mitigate residual output-level mismatch.
- We introduce CD-LoRA, a router-free PEFT framework that achieves strong empirical performance across multi-task benchmarks.

2 Related Work

2.1 Low-Rank Adaptation (LoRA)

LoRA (Hu et al., 2022) is designed to efficiently adapt LLMs by approximating the weight updates $\Delta\mathbf{W}$ of dense matrices within Transformer blocks (Vaswani et al., 2017). Built on the hypothesis that such weight updates exhibit low intrinsic dimensionality, LoRA decomposes $\Delta\mathbf{W}$ into two low-rank matrices, namely LoRA-$\mathbf{A} \in \mathbb{R}^{r\times n}$ and LoRA-$\mathbf{B} \in \mathbb{R}^{m\times r}$, where $\mathbf{W} \in \mathbb{R}^{m\times n}$ denotes the original frozen weight matrix of the model. The rank r is deliberately set to be much smaller than $\min(m, n)$, which reduces the number of trainable parameters from $\mathcal{O}(mn)$ (the order of magnitude for full dense matrix updates) to $\mathcal{O}(r(m+n))$, achieving significant parameter efficiency. For a given input $\mathbf{x}$, the forward propagation process is adjusted as shown:

$$\mathbf{h} = (\mathbf{W} + \Delta\mathbf{W})\mathbf{x} = \mathbf{W}\mathbf{x} + \mathbf{B}\mathbf{A}\mathbf{x}, \quad (1)$$

Here, $\Delta\mathbf{W} = \mathbf{B}\mathbf{A}$ represents the low-rank weight update induced by LoRA. A pivotal advantage of this approach is that post-training, the low-rank update $\Delta\mathbf{W}$ can be seamlessly merged back into the original weight matrix $\mathbf{W}$, resulting in zero additional inference overhead compared to the base model without adaptation.

Several LoRA variants improve adaptation from different perspectives. AdaLoRA (Zhang et al., 2023) adjusts rank budgets across layers, while DoRA (Liu et al., 2024) decomposes updates into magnitude and direction components. PiSSA (Meng et al., 2024) and NLoRA (Guo et al., 2025) further enhance LoRA through improved initialization or alternative low-rank decompositions.

2.2 LoRA for Multi-Task Learning.

Extending LoRA to multi-task learning naturally motivates the introduction of multiple trainable components to accommodate task heterogeneity. Early approaches adopt a **Multi-Adapter** paradigm (Li et al., 2024; Gao et al., 2025), where independent LoRA adapters, specifically separate (B_i, A_i) matrix pairs, are allocated for each task. Though effective, this design causes the number of parameters to grow linearly with the number of tasks, and the strict parameter isolation hinders cross-task knowledge transfer (Huang et al., 2024).

To improve parameter efficiency, subsequent studies (Tian et al., 2024) exploit the asymmetric

Table 1: Architectural Variants of LoRA for Multi-Task Learning

Architecture	Multi-Adapter	Multi-Head	Dynamic Routing
Weight Update	$\Delta W = \sum_{i=1}^{N} B_i A_i$	$\Delta W = \sum_{i=1}^{N} B_i A$	$\Delta W = \sum_{i=1}^{N} \omega_i(x)\, B_i A$

Notation: *$\omega_i(x)$ denotes input-dependent mixing weights, typically produced by a softmax-based gating network.*

roles of LoRA's low-rank factors. Empirical evidence suggests that the down-projection LoRA-A primarily captures task-agnostic and redundant representations, whereas the up-projection LoRA-B encode task-specific variations (Liu et al., 2025). This insight gives rise to the **Multi-Head** architecture, which shares a single projection LoRA-A across tasks while maintaining multiple task-specific heads $\{B_i\}$. Compared to multi-adapter designs, multi-head LoRA substantially reduces parameter overhead while enabling partial knowledge sharing.

Several multi-head LoRA variants (Li et al., 2024; Huang et al., 2024; Gao et al., 2025; Liu et al., 2025) adopt MoE-inspired **Dynamic Routing** to adaptively combine LoRA heads for task specialization. Table 1 summarizes their weight update formulations.

However, input-dependent routing prevents ΔW from being pre-computed and merged into the frozen backbone. Consequently, the router and multiple LoRA heads must be evaluated at inference time, introducing additional latency and memory overhead. This undermines LoRA's key advantage of *zero inference overhead* and raises concerns about **training–inference consistency**.

3 Motivation

3.1 Formalizing the Training–Inference Discrepancy

To examine whether complex routing mechanisms are indeed necessary for multi-task PEFT, we begin by formalizing the mismatch between training and deployment, referred to as the *training–inference discrepancy*.

In routing-based LoRA variants such as R-LoRA (Liu et al., 2025), the training-time output for an input $\mathbf{x} \in \mathbb{R}^n$ under a stochastic dropout mask $\mathbf{z} \sim p(\mathbf{z})$ is given by

$$\Delta W(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{N} \omega_i(\mathbf{x}, \mathbf{z}) \cdot \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \mathbf{z}), \quad (2)$$

where N denotes the number of LoRA heads, $\mathbf{A} \in \mathbb{R}^{r \times n}$ is the shared down-projection matrix, $\mathbf{B}_i \in \mathbb{R}^{m \times r}$ is the up-projection matrix of the i-th head, and $\omega_i(\cdot)$ is a **non-linear** routing weight, typically produced by a **softmax-based gating network**. The operator $\odot$ denotes element-wise multiplication.

At inference time, a deterministic mask $\bar{\mathbf{z}} = \mathbb{E}[\mathbf{z}]$ is used. Due to the **non-linearity** of $\omega_i(\cdot)$, which includes a **softmax operation**, the expected training output does not generally coincide with the inference-time output, i.e., $\mathbb{E}_{\mathbf{z}}[f(\mathbf{x}, \mathbf{z})] \neq f(\mathbf{x}, \bar{\mathbf{z}})$. This mismatch gives rise to a training–inference discrepancy that may degrade predictive stability.

The adapter-level training–inference discrepancy for input $\mathbf{x}$ is defined as

$$\mathrm{TID}_{\mathrm{adp}}^{\mathrm{route}}(\mathbf{x}) = \Bigg\| \mathbb{E}_{\mathbf{z}} \left[\sum_{i=1}^{N} \omega_i(\mathbf{x}, \mathbf{z}) \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \mathbf{z}) \right] - \sum_{i=1}^{N} \omega_i(\mathbf{x}, \bar{\mathbf{z}}) \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \bar{\mathbf{z}}) \Bigg\|, \quad (3)$$

where $\| \cdot \|$ denotes the Frobenius norm.

Discrepancy Decomposition in De-Routed Architectures. To isolate the contribution of routing, we consider a router-free multi-head LoRA variant in which all heads are uniformly weighted. In this case, Equation (3) reduces to

$$\mathrm{TID}_{\mathrm{adp}}^{\mathrm{deroute}}(\mathbf{x}) = \Bigg\| \mathbb{E}_{\mathbf{z}} \left[\sum_{i=1}^{N} \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \mathbf{z}) \right] - \sum_{i=1}^{N} \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \bar{\mathbf{z}}) \Bigg\|. \quad (4)$$

Due to the linearity of $\mathbf{A}$, we have

$$\mathbb{E}_{\mathbf{z}} \left[\sum_{i=1}^{N} \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \mathbf{z}) \right] = \sum_{i=1}^{N} \mathbf{B}_i \mathbf{A}(\mathbf{x} \odot \bar{\mathbf{z}}). \quad (5)$$

Therefore, under the de-routed architecture, the adapter-level training–inference discrepancy is

eliminated:

$$\mathrm{TID}_{\mathrm{adp}}^{\mathrm{deroute}}(\mathbf{x}) = 0. \tag{6}$$

This indicates that removing the router eliminates the adapter-level discrepancy introduced by the nonlinear routing mechanism.

3.2 Empirical Analysis of Representation Structure

To empirically examine the representation behavior suggested by our theoretical analysis, we study the structure of activations produced by the LoRA-A (`down_proj`) module under 12-task joint training. The `down_proj` module corresponds to the shared low-rank projection matrix $\mathbf{A}$, which maps high-dimensional hidden states into a compact intrinsic subspace before task-specific transformations. As shown in Section 3.1, instability in this shared projection is the primary source of the training–inference discrepancy, making it a natural candidate for representation-level analysis.

We extract layer-wise LoRA-A activations from the `down_proj` module and visualize their distributions using t-SNE. This analysis focuses on whether representations induced by different tasks remain separable or collapse into a shared manifold when projected into the low-rank space.

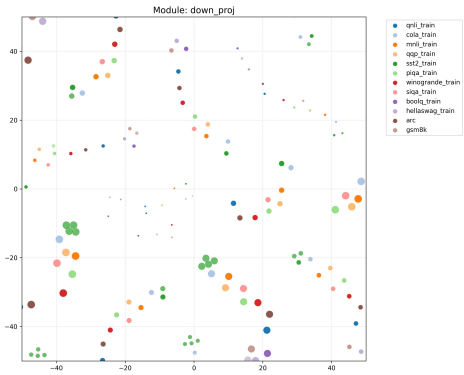

Figure 2: t-SNE visualization of LoRA-A activations in the `down_proj` module. Different colors indicate different tasks. Representations from multiple tasks exhibit substantial overlap in the shared low-rank space.

As illustrated in Figure 2, LoRA-A activations from different tasks are highly intermingled, forming a largely overlapping structure rather than well-separated clusters. Although t-SNE provides only a qualitative perspective, this observation aligns with our theoretical findings: the limitation of multi-task adaptation does not primarily stem from insufficient task partitioning, but from instability within the shared projection space. Detailed visualization settings and extraction procedures are provided in Appendix C.3.

4 Method

4.1 De-Routing and Residual Output-Level Discrepancy

As shown in Equation (4), de-routing eliminates the adapter-level training–inference discrepancy by removing the nonlinear routing mechanism. Specifically, once the input-dependent routing weights are replaced by uniform head aggregation, the remaining adapter transformation becomes linear with respect to the stochastic dropout perturbation, and the adapter-level discrepancy vanishes.

However, eliminating the adapter-level discrepancy does not imply that the entire model is free from training–inference mismatch, since the de-routed adapter output is still propagated through nonlinear downstream Transformer layers. To characterize this residual mismatch, let $g(\mathbf{z})$ denote the composite stochastic output map

$$g(\mathbf{z}) = \Phi_\theta(\mathbf{A}(\mathbf{x} \odot \mathbf{z})) \tag{7}$$

where Φ_θ denotes the downstream nonlinear mapping applied to the stochastic low-rank representation, including subsequent Transformer layers. Although $\mathbf{A}(\mathbf{x} \odot \mathbf{z})$ is linear in $\mathbf{z}$, the composite map $g(\mathbf{z})$ is generally nonlinear due to these downstream transformations.

The output-level training–inference discrepancy is then defined as

$$\mathrm{TID}_{\mathrm{out}}^{\mathrm{deroute}}(\mathbf{x}) = \left\| \mathbb{E}_{\mathbf{z}}\left[g(\mathbf{z})\right] - g(\bar{\mathbf{z}}) \right\|. \tag{8}$$

4.2 Consistency-Driven Alignment in the Shared Low-Rank Space

Output-level Instability Analysis. We perform a second-order Taylor expansion of $g(\mathbf{z})$ around the expectation $\bar{\mathbf{z}}$:

$$\mathbb{E}_{\mathbf{z}}[g(\mathbf{z})] - g(\bar{\mathbf{z}}) \approx \frac{1}{2}\mathrm{Tr}\big(\nabla^2 g(\bar{\mathbf{z}}) \cdot \mathrm{Var}(\mathbf{z})\big). \tag{9}$$

where $\nabla^2 g(\bar{\mathbf{z}})$ is the Hessian matrix of the output-level composite map with respect to the stochastic perturbation $\mathbf{z}$. Here, the curvature comes from the nonlinear downstream mapping Φ_θ, rather than from the linear LoRA-A projection alone. The detailed derivation of Equation (9) is provided in Appendix A.5.

Equation (9) characterizes the residual output-level discrepancy after de-routing. Although the adapter-level discrepancy has been eliminated, dropout-induced variations in the shared low-rank representation can still be amplified by the nonlinear downstream mapping, leading to output-level instability.

From Stochastic Representation Alignment to Consistency Directly optimizing the output-level discrepancy in Equation (8) is computationally prohibitive. Instead, we enforce agreement between two stochastic LoRA-A representations obtained from independent dropout masks, which provides a tractable surrogate for improving output-level consistency.

Specifically, if we model the latent representations $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}$ from two independent forward passes as isotropic Gaussians $\mathcal{P} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2\mathbf{I})$, the symmetric Kullback-Leibler (KL) divergence simplifies to a squared ℓ_2 distance:

$$D_{\mathrm{KL}}(\mathcal{P}_1 \| \mathcal{P}_2) = \frac{1}{2\sigma^2}\|\mathbf{h}^{(1)} - \mathbf{h}^{(2)}\|_2^2, \qquad (10)$$

where $\mathbf{h}^{(j)} = \mathbf{A}(\mathbf{x} \odot \mathbf{z}^{(j)})$ represents the j-th stochastic projection. Minimizing this divergence effectively constrains the spectral norm of $\nabla^2 g(\bar{\mathbf{z}})$, fostering a flatter representation manifold.

The CD-LoRA Objective. Building on this stochastic representation alignment intuition, we propose the Consistency-Driven Alignment (CD-A) mechanism. For each input $\mathbf{x}$, we perform two independent forward passes through the shared LoRA-A matrix using distinct dropout masks $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$,

yielding two latent realizations $\mathbf{h}^{(1)} = \mathbf{A}(\mathbf{x} \odot \mathbf{z}^{(1)})$ and $\mathbf{h}^{(2)} = \mathbf{A}(\mathbf{x} \odot \mathbf{z}^{(2)})$. To mitigate the residual output-level TID, we define the symmetric KL divergence as $D_{\mathrm{SKL}}(P\|Q) = D_{\mathrm{KL}}(P\|Q) + D_{\mathrm{KL}}(Q\|P)$. The consistency loss is then simplified as:

$$\mathcal{L}_{\mathrm{cons}} = \frac{1}{2} D_{\mathrm{SKL}}\big(P(\cdot|\mathbf{x}, \mathbf{h}^{(1)}) \,\|\, P(\cdot|\mathbf{x}, \mathbf{h}^{(2)})\big), \qquad (11)$$

Notations: *$\mathbf{h}^{(j)}$ denotes the j-th latent realization from LoRA-A; $P(\cdot|\mathbf{x}, \mathbf{h}^{(j)})$ is the corresponding output distribution.*

The final training objective of CD-LoRA integrates the task-specific negative log-likelihood (NLL) with the consistency regularization:

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{task}} + \lambda\, \mathcal{L}_{\mathrm{cons}}, \qquad (12)$$

Notations: *λ is a balancing hyperparameter governing the intensity of the consistency constraint; $\mathcal{L}_{\mathrm{task}}$ denotes the primary empirical risk across the multi-task suite, ensuring task-specific predictive proficiency; $\mathcal{L}_{\mathrm{cons}}$ serves as a stochastic consistency regularizer to mitigate the residual output-level TID after de-routing.*

To facilitate understanding, we summarize the complete derivation of CD-LoRA in Appendix A.

Since the input $\mathbf{x}$ is shared across both passes, the consistency loss $\mathcal{L}_{\mathrm{cons}}$ explicitly aligns stochastic representations within the LoRA-$\mathbf{A}$ module. This alignment reduces dropout-induced variance in the shared low-rank space and improves the stability of the downstream output map.

As illustrated in Figure 4, CD-LoRA removes unstable routers and adopts a dual-pass design: $\mathbf{A}$ serves as the primary projection, while $\mathbf{A}'$ provides an auxiliary stochastic view. The consistency loss enforces agreement between these views, thereby mitigating residual output-level training–inference mismatch after de-routing.

5 Experiments

We evaluate CD-LoRA against representative LoRA-based multi-task fine-tuning methods. Unless otherwise specified, all implementation details and hyperparameter settings are deferred to the Appendix C.4.

5.1 Experimental Setup

Backbone Models. We conduct experiments on five decoder-only LLMs, including **Qwen2.5-3B**, **Qwen2.5-7B** and **Qwen2.5-14B** (Hui et al., 2024), as well as **Llama2-7B** (Touvron et al., 2023) and **Llama3-8B** (Team, 2024).

Baselines. We compare CD-LoRA with widely used parameter-efficient multi-task adaptation methods, including **LoRA** (Hu et al., 2022), **HydraLoRA** (Tian et al., 2024), **R-LoRA** (Liu et al., 2025). All methods are implemented under identical optimization settings, batch sizes, and training schedules to ensure a fair comparison.

Evaluation Protocol. All models are jointly fine-tuned and evaluated on a unified benchmark comprising the **12 tasks** listed in Appendix C.1.2, covering natural language inference, commonsense reasoning, sentiment analysis, and mathematical reasoning. Task performance is reported using standard accuracy, and overall performance is summarized by the average across tasks.

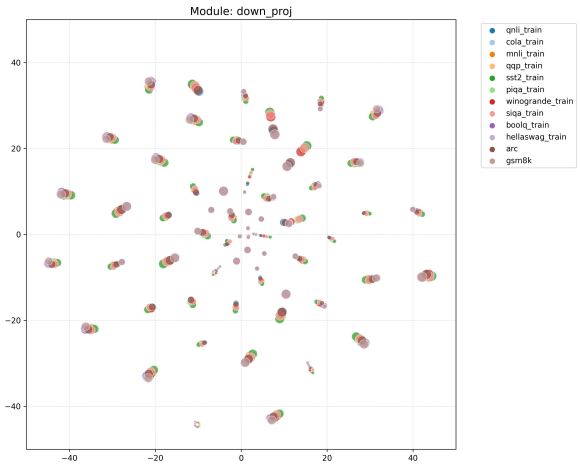

Figure 3: t-SNE visualization of LoRA-A activations learned by CD-LoRA. Different colors correspond to different tasks. CD-LoRA exhibits compact and well-separated clusters, indicating improved task-level representation consistency.

5.2 Main Results on Multi-Task Benchmarks

Table 6 reports the performance of all methods on Qwen2.5-7B and Qwen2.5-14B. Results on Llama2-7B and Llama3-8B are provided in the Appendix C.4.1.

Discussion. Experimental results across both backbones demonstrate that CD-LoRA consistently achieves superior average performance while maintaining competitive memory efficiency.

On the Qwen2.5-7B backbone, CD-LoRA achieves the highest **Avg.** score of **83.92%**, outperforming the routing-based R-LoRA (83.12%) by **0.80%**. Notably, while achieving superior performance, CD-LoRA's peak memory usage (56,882 MB) is significantly lower than that of routing-based HydraLoRA (87,860 MB), representing a **35.3% reduction** in memory overhead. The trend scales positively with model capacity: on Qwen2.5-14B, CD-LoRA further widens the gap, reaching **85.90% Avg.**, a **1.18% improvement** over R-LoRA.

5.3 Analysis of LoRA-A Subspace Consistency

Under the same conditions as in Section 3.2, we visualize the LoRA-A representations of CD-LoRA, as shown in Figures 3 and 2. CD-LoRA yields significantly more compact and task-separable embeddings, indicating that enforcing consistency in the shared low-rank space promotes stable and discriminative task representations—validating the effectiveness of our consistency-driven approach.

6 Ablation Studies

6.1 Module-level Ablation on 7B Models

To disentangle the individual and combined effects of routing mechanisms and consistency alignment, we conduct a controlled module-level ablation on the Qwen2.5-7B backbone across 5 tasks. As summarized in Table 3, we construct a 2×2 ablation matrix by independently enabling or disabling (i) dynamic routing and (ii) explicit consistency alignment.

Efficacy of De-routing. Our results reveal that simply removing the dynamic routing mechanism (DR-LoRA vs. R-LoRA) yields a significant performance boost, elevating the **Avg.** score from **87.59%** to **88.29%** while simultaneously reducing peak memory usage by **24.5%** (from 66,071 MB to 49,869 MB). This substantial gain suggests that routing-induced non-linearity introduces detrimental optimization overhead rather than task-specific benefits. This empirical evidence directly supports our theoretical formalization in Section 3.1, confirming that the non-linear training–inference discrepancy serves as a performance bottleneck.

Synergy between Alignment and De-routing. The introduction of consistency-driven alignment further enhances the de-routed architecture. **CD-LoRA** (the full model) achieves the highest performance across all evaluated tasks, reaching an **Avg.** of **88.56%**. Notably, while adding consistency alignment to a routing-based model (CR-LoRA) provides a marginal improvement of **0.32%** over R-LoRA, it incurs a significant **22.5%** memory penalty. In contrast, CD-LoRA achieves superior results with far greater efficiency, indicating that consistency regularization is most effective when the structural instability of routers is eliminated.

[3]† **R-LoRA**: Baseline multi-task LoRA using non-linear routing mechanisms.

[4]‡ **CR-LoRA**: **R**-LoRA variant incorporating our **C**onsistency alignment module without removing the router.

[5]∗ **DR-LoRA**: **D**e-**R**outed LoRA, a simplified parallel multi-head structure without routing or Consistency constraints.

[6]§ **CD-LoRA**: Our proposed **C**onsistency-**D**riven LoRA, combining the de-routed architecture with explicit consistency alignment.

Table 2: Performance comparison of multi-task fine-tuning methods on Qwen2.5-7B and Qwen2.5-14B.

Method	QNLI	CoLA	MNLI	QQP	SST-2	PIQA	Wino	SIQA	BoolQ	Hella	ARC	GSM8K	Avg.	Mem∗	Thr.†
Qwen2.5-7B															
LoRA	82.40	84.30	87.20	86.40	95.44	88.30	78.70	60.30	88.10	94.20	91.34	48.50	82.10	60062	11.34
HydraLoRA	83.70	85.20	88.10	87.30	96.33	89.70	79.20	60.70	88.60	95.30	91.60	**51.60**	83.11	87860	7.374
R-LoRA	83.70	85.40	88.50	86.90	96.44	90.10	79.20	61.00	88.90	95.20	91.83	50.30	83.12	66070	8.375
CD-LoRA	**84.60**	**85.70**	**89.60**	**87.80**	**96.79**	**90.20**	**83.50**	**61.40**	**89.60**	95.30	92.17	50.40	**83.92**	61082	9.150
Qwen2.5-14B															
LoRA	83.50	85.90	88.70	87.30	96.33	90.40	79.50	60.90	89.00	95.70	92.56	51.60	83.45	73691	9.661
HydraLoRA	85.10	86.40	89.40	86.80	96.63	91.70	85.70	61.10	90.37	96.40	93.21	53.30	84.68	107796	4.097
R-LoRA	85.20	86.40	**89.80**	86.40	96.67	91.77	84.70	61.00	90.71	96.80	93.33	**53.90**	84.72	81062	4.662
CD-LoRA	**87.30**	**87.30**	89.50	**87.40**	96.67	**92.63**	**89.60**	**63.90**	**91.59**	**97.10**	**93.90**	**53.90**	**85.90**	74870	5.310

∗ **Mem**: Peak GPU memory usage, measured in MB under identical hardware and training settings.

† **Thr.**: Training throughput, measured in samples/s under identical hardware and training settings.

Table 3: Module-level ablation on 5 tasks using 7B models.

Method	Router	Consistency	QNLI	MNLI	Wino	BoolQ	ARC	Avg.	Mem∗
R-LoRA†	✓	✗	84.30	90.30	83.00	88.40	91.94	87.59	66071
CR-LoRA‡	✓	✓	84.20	90.20	83.90	89.10	92.17	87.91	80943
DR-LoRA∗	✗	✗	85.10	90.60	84.60	89.00	92.17	88.29	49869
CD-LoRA§	✗	✓	**85.40**	**90.90**	**85.00**	**89.20**	**92.29**	**88.56**	56883

Table 4: Generalization performance on the BBH benchmark after 12-task training.

Method	Router	Cons.	BBH Avg.	BBH Ov.	Mem (MB)
R-LoRA†	✓	✗	51.70	50.77	66070
CR-LoRA‡	✓	✓	52.50	51.69	80942
DR-LoRA∗	✗	✗	52.07	51.27	**45510**
CD-LoRA§	✗	✓	**53.31**	**52.58**	60288

6.2 Generalization to BBH Reasoning Benchmark

To further examine whether CD-LoRA generalizes beyond the main 12 tasks, we evaluate the models on the BBH benchmark after training. Details of BBH are provided in Appendix C.2.

As shown in Table 4, CD-LoRA achieves the best generation-style reasoning performance, with **53.31** BBH Avg. and **52.58** BBH Ov. Compared with R-LoRA, CD-LoRA improves BBH Avg. by **1.61** points while reducing memory usage from 66070 MB to 60288 MB. Although DR-LoRA has the lowest memory cost, its BBH performance is lower than CD-LoRA, indicating that de-routing alone is insufficient. Compared with CR-LoRA, CD-LoRA obtains better accuracy with substantially lower memory, suggesting that consistency alignment is more effective when the routing mechanism is removed. These results show that CD-LoRA generalizes beyond the main training setting and transfers effectively to generation-style reasoning tasks.

6.3 Mechanistic Analysis: Subspace Coherence in LoRA-B

To uncover the mechanistic origins of CD-LoRA's robust multi-task generalization, we conduct a fine-grained analysis of the learned LoRA-B matrices on the Qwen2.5-14B backbone. At this scale, representational divergence across tasks and functional modules becomes particularly pronounced.
We focus on three critical Transformer components:
`up_proj`: Expands input features into a high-dimensional latent space.
`gate_proj`: Modulates feature flow via nonlinear gating.
`down_proj`: Projects the gated representations back to the residual dimension.

We quantify the coherence of the learned low-rank subspaces by calculating the average pairwise cosine similarity $\mathcal{S}$ between task-specific LoRA-$\mathbf{B}$. For any two tasks i and j in a given layer, the similarity is defined as:

$$\cos(\mathbf{B}_i, \mathbf{B}_j) = \frac{\text{vec}(\mathbf{B}_i)^\top \text{vec}(\mathbf{B}_j)}{\|\text{vec}(\mathbf{B}_i)\|_2 \|\text{vec}(\mathbf{B}_j)\|_2} \quad (13)$$

where $\text{vec}(\cdot)$ denotes the flattening operation and $\|\cdot\|_2$ is the ℓ_2 norm. To evaluate the global alignment of a functional module (e.g., `gate_proj`), we

report the mean similarity $\bar{\mathcal{S}}$ across all L layers:

$$\bar{\mathcal{S}} = \frac{1}{L}\sum_{l=1}^{L}\left(\frac{2}{K(K-1)}\sum_{1\le i<j\le K}\cos(\mathbf{B}_{l,i},\mathbf{B}_{l,j})\right) \tag{14}$$

where $L = 48$ for the 14B model and K denotes the number of heads in the multi-head architecture.

Quantitative Analysis. As summarized in Table 5, routing-based methods (R-LoRA and CR-LoRA) exhibit significant *subspace fragmentation*. R-LoRA achieves a mean similarity of only **77.64%**, dropping further to **74.80%** in the `gate_proj`. This suggests that non-linear routing prompts tasks to converge into isolated, heterogeneous parameter regions. While de-routing (DR-LoRA) increases the average similarity to **82.07%** by enforcing shared parameterization, it still lacks explicit cross-task coordination.

Notably, CD-LoRA achieves the highest similarity consistently across all projections, with a global average of **84.93%**. In the most challenging `gate_proj`, CD-LoRA improves the similarity from R-LoRA's 74.80% to **84.49%** (**+9.69%** absolute gain). This uniform enhancement indicates that consistency-driven alignment effectively stabilizes the representation manifold across diverse Transformer operators.

Qualitative Visualization. Figure 5 illustrates the layer-wise cosine similarity of CD-LoRA across different projection layers. In contrast to the erratic fluctuations or fragmented representations typical of routing-based approaches (R-LoRA in Figure 6 and CR-LoRA in Figure 7), CD-LoRA maintains a high and stable similarity mean across nearly all layers. This cross-layer consistency qualitatively reinforces our findings: the consistency-driven alignment mechanism effectively constrains the low-rank subspaces of diverse tasks within a coherent manifold, thereby preventing representational drift during multi-task adaptation in large-scale models.

Discussion. These results provide direct mechanistic evidence that the performance gains of CD-LoRA stem from **subspace coherence**. By eliminating routing-induced fragmentation and explicitly aligning task representations, CD-LoRA enforces a shared inductive bias that prevents weight drifting. This structural stability becomes increasingly critical as model capacity and task complexity scale.

Table 5: Quantification of $\mathbf{B}$ matrix similarity (%) across projection layers for Qwen2.5-14B.

Method	up_proj	down_proj	gate_proj	Avg.
R-LoRA†	77.98	80.15	74.80	77.64
CR-LoRA‡	79.30	82.00	75.23	78.84
DR-LoRA*	82.20	82.50	81.52	82.07
CD-LoRA§	**85.03**	**85.26**	**84.49**	**84.93**

6.4 Sensitivity to the Consistency Weight λ

The hyperparameter λ in Equation (12) controls the strength of the consistency-driven alignment objective $\mathcal{L}_{\text{cons}}$. We evaluate its sensitivity on Qwen2.5-3B, Qwen2.5-7B, and Qwen2.5-14B across the 12-task evaluation suite.

As shown in Appendix E, CD-LoRA is generally robust to the choice of λ across different model scales. The best performance is achieved at $\lambda = 0.03$ on Qwen2.5-3B, around $\lambda = 0.01$ on Qwen2.5-7B, and $\lambda = 0.05$ on Qwen2.5-14B. Across all three scales, moderate consistency weights provide stable gains over the corresponding baselines, while overly large values such as 0.5 or 1.0 may over-constrain the shared LoRA-$\mathbf{A}$ subspace, with the degradation being more evident on the smaller Qwen2.5-3B backbone.

7 Conclusion

This work provides a two-level analysis of training–inference discrepancy in router-based multi-head LoRA. We show that nonlinear routing induces an adapter-level discrepancy between training and inference, while residual output-level mismatch may still arise after de-routing due to downstream nonlinear transformations. Based on this analysis, we introduce **CD-LoRA**, a router-free framework that combines de-routing with consistency-driven alignment in a shared low-rank space. De-routing eliminates the adapter-level routing discrepancy, while consistency alignment mitigates residual output-level mismatch by enforcing agreement between stochastic low-rank representations.

Empirical results show that CD-LoRA consistently outperforms strong baselines such as R-LoRA while reducing routing-related computational overhead. Our findings suggest that collaborative learning within a coherent shared low-rank space offers a more effective alternative to explicit expert partitioning. By challenging the necessity of dynamic routing in multi-task LoRA, this work establishes a simple and effective paradigm for robust PEFT.

Limitations

While CD-LoRA yields promising results, certain limitations remain. Despite rigorous experimental validation, the complexity of multi-task PEFT invites broader evaluation. Our current scope focuses on NLP; thus, adapting CD-LoRA to computer vision, speech, and multimodal settings represents a crucial next step. These research directions are essential to harness the model's full capabilities and to generalize its impact across a wider technological landscape.

Although CD-LoRA improves the efficiency and stability of multi-task fine-tuning, it also lowers the computational barrier for adapting large language models to harmful applications. Because this method facilitates cross-task knowledge transfer via a shared low-rank space, biases, spurious correlations, or unsafe patterns in the training data may propagate across tasks. Furthermore, enhanced efficiency could accelerate deployment in resource-constrained environments lacking adequate safety evaluations or human oversight. Therefore, we strongly recommend conducting task-specific safety audits, bias assessments, and continuous monitoring before deploying CD-LoRA, particularly in high-stakes domains.

References

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. Piqa: Reasoning about physical commonsense in natural language. *Preprint*, arXiv:1911.11641.

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, and 12 others. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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. *Preprint*, arXiv:2110.14168.

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

Chenlu Guo, Yi Chang, and Yuan Wu. 2025. NLoRA: Nyström-initiated low-rank adaptation for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 1371–1385, Suzhou, China. Association for Computational Linguistics.

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. 2024. Lorahub: Efficient cross-task generalization via dynamic lora composition. *Preprint*, arXiv:2307.13269.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, and 5 others. 2024. Qwen2.5-Coder Technical Report. *arXiv e-prints*, arXiv:2409.12186.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

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. *Preprint*, arXiv:2404.15159.

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

Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. 2023. Scaling down to scale up: A guide to parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.15647*.

Jinda Liu, Yi Chang, and Yuan Wu. 2025. R-lora: Random initialization of multi-head lora for multi-task learning. *arXiv preprint arXiv:2502.15455*.

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 *Forty-first International Conference on Machine Learning*.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Llama Team. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

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.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Haowen Wang, Tao Sun, Kaixiang Ji, Jian Wang, Cong Fan, and Jinjie Gu. 2024. Orchmoe: Efficient multi-adapter learning with task-skill synergy. *arXiv preprint arXiv:2401.10559*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

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

A Derivation of CD-LoRA

In this appendix, we provide a complete derivation of CD-LoRA using the same notation as the main text. We first show that nonlinear routing induces an adapter-level training–inference discrepancy, and that de-routing eliminates this discrepancy. We then analyze the residual output-level discrepancy after de-routing and introduce consistency-driven alignment as a tractable surrogate for mitigating this residual mismatch.

A.1 Preliminaries and Notation

Let $x \in \mathbb{R}^n$ denote the input activation to a linear layer, and let $z \sim p(z)$ denote the stochastic dropout mask used during training. At inference time, the stochastic mask is replaced by its expectation:

$$\bar{z} = \mathbb{E}[z], \tag{15}$$

with covariance

$$\mathrm{Var}(z) = \mathbb{E}\left[(z-\bar{z})(z-\bar{z})^\top\right]. \tag{16}$$

Following the main text, let $A \in \mathbb{R}^{r\times n}$ denote the shared LoRA-A down-projection matrix, and let $B_i \in \mathbb{R}^{m\times r}$ denote the i-th LoRA-B up-projection matrix. The number of LoRA heads is denoted by N.

In routing-based multi-head LoRA, the training-time adapter output under dropout is written as

$$\Delta W_{\mathrm{route}}(x,z) = \sum_{i=1}^{N} \omega_i(x,z) B_i A(x \odot z), \tag{17}$$

where $\omega_i(x,z)$ denotes the input-dependent routing weight, typically produced by a softmax-based gating network, and $\odot$ denotes element-wise multiplication.

A.2 Adapter-level Training–Inference Discrepancy

For routing-based LoRA, the adapter-level training–inference discrepancy is defined as

$$\mathrm{TID}_{\mathrm{adp}}^{\mathrm{route}}(x) = \left\| \mathbb{E}_z\left[\sum_{i=1}^{N} \omega_i(x,z) B_i A(x\odot z)\right] - \sum_{i=1}^{N} \omega_i(x,\bar{z}) B_i A(x\odot \bar{z}) \right\|. \tag{18}$$

Because the routing weights are generated by a nonlinear gating function, the expectation cannot generally be moved through $\omega_i(\cdot)$. Therefore,

$$\mathbb{E}_z\left[\Delta W_{\mathrm{route}}(x,z)\right] \neq \Delta W_{\mathrm{route}}(x,\bar{z}), \tag{19}$$

which yields a nonzero adapter-level discrepancy.

A.3 De-routing Eliminates Adapter-level Discrepancy

To isolate the contribution of routing, CD-LoRA removes the input-dependent router and aggregates all LoRA heads directly. The de-routed adapter output is

$$\Delta W_{\mathrm{deroute}}(x,z) = \sum_{i=1}^{N} B_i A(x\odot z). \tag{20}$$

The corresponding adapter-level discrepancy becomes

$$\mathrm{TID}_{\mathrm{adp}}^{\mathrm{deroute}}(x) = \left\| \mathbb{E}_z\left[\sum_{i=1}^{N} B_i A(x\odot z)\right] - \sum_{i=1}^{N} B_i A(x\odot \bar{z}) \right\|. \tag{21}$$

Since A and B_i are linear maps, we have

$$\begin{aligned}\mathbb{E}_z\left[\sum_{i=1}^{N} B_i A(x\odot z)\right] &= \sum_{i=1}^{N} B_i A\left(x \odot \mathbb{E}[z]\right) \\ &= \sum_{i=1}^{N} B_i A(x\odot \bar{z}).\end{aligned} \tag{22}$$

Substituting Eq. (22) into Eq. (21) gives

$$\mathrm{TID}_{\mathrm{adp}}^{\mathrm{deroute}}(x) = 0. \tag{23}$$

Thus, de-routing eliminates the adapter-level discrepancy induced by the nonlinear routing mechanism.

A.4 Residual Output-level Discrepancy after De-routing

Although de-routing removes the adapter-level discrepancy, the entire model may still suffer from training–inference mismatch because the de-routed adapter output is propagated through nonlinear downstream Transformer layers.

Following the main text, let

$$g(z) = \Phi_\theta\left(A(x\odot z)\right), \tag{24}$$

where Φ_θ denotes the downstream nonlinear mapping applied to the stochastic low-rank representation. Although $A(x\odot z)$ is linear in z, the composite map $g(z)$ is generally nonlinear because of Φ_θ.

The residual output-level training–inference discrepancy is defined as

$$\mathrm{TID}_{\mathrm{out}}^{\mathrm{deroute}}(x) = \left\|\mathbb{E}_z[g(z)] - g(\bar{z})\right\|. \tag{25}$$

Unlike the adapter-level discrepancy, this output-level discrepancy does not necessarily vanish.

A.5 Second-order Taylor Characterization

We characterize the residual output-level discrepancy using a second-order Taylor expansion around $\bar{z}$. Since $g(z)$ can be vector-valued, we apply the expansion coordinate-wise. Let $g_k(z)$ denote the k-th coordinate of $g(z)$. Then

$$\begin{aligned} g_k(z) \approx\ & g_k(\bar{z}) + \nabla g_k(\bar{z})^\top (z - \bar{z}) \\ & + \frac{1}{2}(z - \bar{z})^\top \nabla^2 g_k(\bar{z})(z - \bar{z}). \end{aligned} \tag{26}$$

Taking expectation over z, the first-order term vanishes because $\mathbb{E}_z[z - \bar{z}] = 0$. For the second-order term, we have

$$\begin{aligned} & \mathbb{E}_z \left[\frac{1}{2}(z - \bar{z})^\top \nabla^2 g_k(\bar{z})(z - \bar{z}) \right] \\ & = \frac{1}{2} \mathrm{Tr} \left(\nabla^2 g_k(\bar{z}) \cdot \mathrm{Var}(z) \right). \end{aligned} \tag{27}$$

Therefore,

$$\mathbb{E}_z[g_k(z)] - g_k(\bar{z}) \approx \frac{1}{2} \mathrm{Tr} \left(\nabla^2 g_k(\bar{z}) \cdot \mathrm{Var}(z) \right). \tag{28}$$

Equivalently, using the compact notation in the main text,

$$\mathbb{E}_z[g(z)] - g(\bar{z}) \approx \frac{1}{2} \mathrm{Tr} \left(\nabla^2 g(\bar{z}) \cdot \mathrm{Var}(z) \right). \tag{29}$$

This shows that the residual output-level mismatch is governed by the curvature of the downstream composite map g and the variance of the stochastic dropout perturbation.

A.6 Consistency-driven Alignment

Directly optimizing Eq. (25) is computationally expensive. CD-LoRA therefore introduces consistency-driven alignment as a tractable surrogate. Specifically, we sample two independent dropout masks $z^{(1)}, z^{(2)} \sim p(z)$ and obtain two stochastic LoRA-A representations:

$$h^{(1)} = A(x \odot z^{(1)}), \qquad h^{(2)} = A(x \odot z^{(2)}). \tag{30}$$

For intuition, we model the two latent representations as isotropic Gaussian distributions with shared covariance:

$$\mathcal{P}_1 = \mathcal{N}(h^{(1)}, \sigma^2 I), \qquad \mathcal{P}_2 = \mathcal{N}(h^{(2)}, \sigma^2 I). \tag{31}$$

Under this assumption, the KL divergence has the closed form

$$D_{\mathrm{KL}}(\mathcal{P}_1 \| \mathcal{P}_2) = \frac{1}{2\sigma^2} \left\| h^{(1)} - h^{(2)} \right\|_2^2. \tag{32}$$

The symmetric KL divergence is

$$D_{\mathrm{SKL}}(\mathcal{P}_1 \| \mathcal{P}_2) = D_{\mathrm{KL}}(\mathcal{P}_1 \| \mathcal{P}_2) + D_{\mathrm{KL}}(\mathcal{P}_2 \| \mathcal{P}_1), \tag{33}$$

which reduces to

$$D_{\mathrm{SKL}}(\mathcal{P}_1 \| \mathcal{P}_2) = \frac{1}{\sigma^2} \left\| h^{(1)} - h^{(2)} \right\|_2^2. \tag{34}$$

Thus, minimizing the symmetric KL divergence between two stochastic views is equivalent to reducing their squared distance in the shared low-rank space.

In practice, CD-LoRA applies the consistency constraint to the corresponding output distributions:

$$\mathcal{L}_{\mathrm{cons}} = \frac{1}{2} D_{\mathrm{SKL}} \left(P(\cdot|x, h^{(1)}) \| P(\cdot|x, h^{(2)}) \right), \tag{35}$$

where

$$D_{\mathrm{SKL}}(P \| Q) = D_{\mathrm{KL}}(P \| Q) + D_{\mathrm{KL}}(Q \| P). \tag{36}$$

A.7 Connection to the Output-level Residual

We now relate the consistency objective to the output-level residual in Eq. (29). Assume that Φ_θ is locally Lipschitz continuous around the stochastic low-rank representations, with constant L_Φ. Then

$$\begin{aligned} \left\| g(z^{(1)}) - g(z^{(2)}) \right\| &= \left\| \Phi_\theta(h^{(1)}) - \Phi_\theta(h^{(2)}) \right\| \\ &\leq L_\Phi \left\| h^{(1)} - h^{(2)} \right\|. \end{aligned} \tag{37}$$

Eq. (37) shows that reducing the discrepancy between two stochastic LoRA-A representations limits the induced output variation under the downstream nonlinear mapping. Since Eq. (34) shows that the symmetric KL surrogate is proportional to $\|h^{(1)} - h^{(2)}\|_2^2$, minimizing $\mathcal{L}_{\mathrm{cons}}$ reduces stochastic variation in the shared low-rank space and consequently stabilizes the output map.

Therefore, although $\mathcal{L}_{\mathrm{cons}}$ does not directly optimize the Taylor residual

$$\frac{1}{2} \mathrm{Tr} \left(\nabla^2 g(\bar{z}) \cdot \mathrm{Var}(z) \right), \tag{38}$$

it serves as a tractable surrogate for mitigating the residual output-level mismatch. By aligning stochastic low-rank representations before they are propagated through Φ_θ, CD-LoRA reduces dropout-induced fluctuations that may otherwise be amplified by downstream nonlinear transformations.

A.8 Final Objective

The final CD-LoRA training objective combines the task-specific negative log-likelihood with the consistency-driven alignment loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{cons}}, \tag{39}$$

where $\mathcal{L}_{\text{task}}$ denotes the primary task loss, $\mathcal{L}_{\text{cons}}$ denotes the consistency regularization term, and λ controls the strength of consistency-driven alignment.

A.9 Summary

The derivation establishes four points. First, nonlinear routing induces an adapter-level training–inference discrepancy because the stochastic dropout expectation does not commute with input-dependent routing weights. Second, de-routing eliminates this adapter-level discrepancy because the remaining adapter transformation is linear with respect to the dropout mask. Third, a residual output-level discrepancy may still remain because the low-rank representation is propagated through nonlinear downstream transformations. Finally, consistency-driven alignment mitigates this residual mismatch by aligning stochastic low-rank representations in the shared LoRA-A space and thereby stabilizing the downstream output map.

B Overview of the CD-LoRA framework

Figure 4 presents the architecture of CD-LoRA, highlighting its router-free design and the mechanism for consistency alignment.

C Additional Experimental Details

This appendix provides additional details on datasets, training configurations, and implementation specifics of CD-LoRA, complementing the main paper and facilitating reproducibility.

C.1 Experimental Datasets

C.1.1 5 tasks Setting

1. **QNLI** (Wang et al., 2018):
   Question-answering Natural Language Inference.

2. **MNLI** (Williams et al., 2018):
   Multi-Genre Natural Language Inference

3. **Winogrande** (Sakaguchi et al., 2021):
   Commonsense coreference resolution.

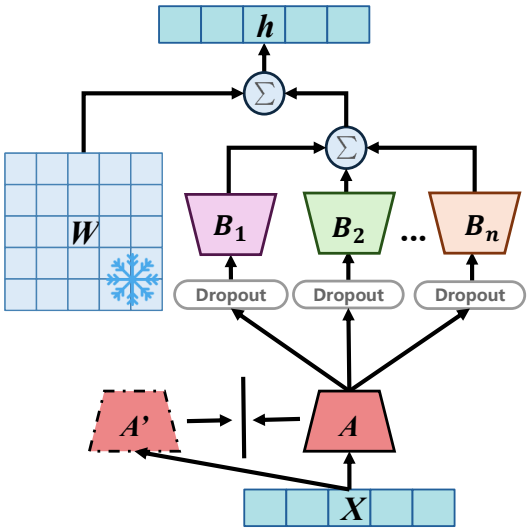

Figure 4: Overview of the CD-LoRA framework. CD-LoRA removes unstable routers and adopts a dual-pass design: A serves as the primary projection, while A' provides an auxiliary stochastic view.

4. **BoolQ** (Clark et al., 2019):
   Boolean Questions.

5. **ARC** (Clark et al., 2018):
   AI Reasoning Challenge.

These tasks jointly cover entailment, commonsense reasoning, and multi-choice question answering, allowing us to examine cross-task interference under a moderate training scale.

C.1.2 12 tasks Setting

1. **QNLI** (Wang et al., 2018):
   Question-answering Natural Language Inference.

2. **CoLA** (Warstadt et al., 2019):
   Corpus of Linguistic Acceptability.

3. **MNLI** (Williams et al., 2018):
   Multi-Genre Natural Language Inference.

4. **QQP** (Wang et al., 2018):
   Quora Question Pairs.

5. **SST-2** (Socher et al., 2013):
   Stanford Sentiment Treebank.

6. **PIQA** (Bisk et al., 2019):
   Physical Interaction Question Answering.

7. **Winogrande** (Sakaguchi et al., 2021):
   Commonsense coreference resolution.

8. **SIQA** (Sap et al., 2019):
   Social Interaction Question Answering.

9. **BoolQ** (Clark et al., 2019): Boolean Questions.
10. **HellaSwag** (Zellers et al., 2019): Adversarial commonsense inference with long contexts.
11. **ARC** (Clark et al., 2018): AI Reasoning Challenge.
12. **GSM8K** (Cobbe et al., 2021): Grade School Math reasoning benchmark.

This setting significantly increases task heterogeneity, especially by introducing mathematical reasoning (GSM8K) and fine-grained linguistic acceptability judgments (CoLA), making it well suited for stress-testing multi-adapter coordination.

C.2 BBH Benchmark

We additionally evaluate CD-LoRA on the BIG-Bench Hard (BBH) benchmark (Suzgun et al., 2023), which consists of a curated subset of challenging tasks from BIG-Bench that are known to be difficult for LLMs without explicit reasoning supervision.

BBH focuses on tasks that require multi-step reasoning, compositional generalization, and robust instruction following. The benchmark spans diverse reasoning categories, including logical deduction, algorithmic reasoning, commonsense inference, and symbolic manipulation, making it particularly well suited for assessing the generalization and coordination ability of parameter-efficient adaptation methods.

Following prior work, we adopt the standard BBH evaluation protocol and report accuracy averaged across all selected tasks. For models trained in the 12-task multi-task setting, BBH evaluation is performed in a zero-shot manner using task-specific prompts provided by the benchmark. No BBH data is used during training.

In Table 4, we report both the average performance across BBH tasks (BBH Avg.) and the overall accuracy aggregated over all examples (BBH Ov.). Memory consumption is measured as the peak GPU memory usage during training, reflecting the practical efficiency of different adaptation methods.

C.3 Visualization Setup for Representation Analysis

This section describes the experimental setup used for the representation visualization in Section 3.2.

Model and Training Setting. All representations are extracted from a Qwen2.5-14B backbone equipped with multi-task LoRA adapters trained on a 12-task C.1.2 mixture. The LoRA configuration follows the main experimental setting, with rank $r = 4$ and adapters inserted into the feed-forward network (FFN) projection layers. Unless otherwise specified, all experiments are conducted in evaluation mode without gradient updates.

Target Module. We focus on the LoRA-A matrix corresponding to the `down_proj` module in the FFN. This module implements the shared low-rank down-projection and serves as the common feature bottleneck across tasks. As shown in Section 3.1, instability in this shared projection is the dominant source of the training–inference discrepancy, motivating its use for representation analysis.

Activation Extraction. For each transformer layer, we register forward hooks on the LoRA-A (`down_proj`) module and extract its input activations. Given an input sequence, token-level activations are averaged across the sequence length to obtain a single representation vector per layer. All representations are detached from the computation graph and collected on the CPU for subsequent analysis.

Task Sampling. For each task in the 12-task C.1.2 training set, we uniformly sample a small number of input examples (one example per task in the main analysis) to avoid bias toward high-resource tasks. Representations are extracted from all transformer layers for each sampled input.

Dimensionality Reduction and Visualization. The collected representations are projected to two dimensions using t-SNE with PCA initialization and a fixed random seed for reproducibility. All visualizations use identical axis limits and aspect ratios to ensure consistent geometric interpretation across settings. Different colors denote different tasks, while marker sizes encode layer depth.

C.4 Main Experiment

C.4.1 Performance on Llama2-7B and Llama3-8B

Table 6 evaluates the performance of CD-LoRA on Llama2-7B and Llama3-8B models.

C.4.2 Model and Training Configuration

All experiments are conducted on top of the Qwen2.5 family of pretrained causal language mod-

Table 6: Performance comparison of multi-task fine-tuning methods on Llama2-7B and Llama3-8B.

Method	QNLI	CoLA	MNLI	QQP	SST-2	PIQA	Wino	SIQA	BoolQ	Hella	ARC	GSM8K	Avg.	Mem∗	Thr.
Llama2-7B															
LoRA	80.20	84.00	83.90	84.30	95.07	79.20	68.30	51.90	85.83	89.20	71.00	25.40	74.86	49166	13.71
HydraLoRA	80.70	82.70	84.30	84.70	95.07	79.09	69.90	51.90	86.39	89.80	71.00	23.90	74.95	86164	8.88
R-LoRA	80.00	83.20	84.20	84.80	94.84	79.31	69.70	51.50	85.77	89.60	70.10	24.50	74.79	60472	9.41
CD-LoRA	**82.50**	**86.30**	**84.90**	**85.10**	**96.44**	**80.97**	**75.20**	**55.30**	86.51	**91.80**	**76.90**	22.10	**77.00**	52457	10.87
Llama3-8B															
LoRA	84.00	85.70	86.30	86.20	96.22	86.95	82.40	58.20	**90.35**	**94.30**	87.11	**53.00**	82.56	51133	11.92
HydraLoRA	83.20	85.60	87.50	86.70	**96.56**	87.17	81.90	58.20	89.85	93.70	87.00	52.00	82.45	93669	7.40
R-LoRA	83.50	**86.40**	86.70	86.60	**96.56**	**88.05**	81.80	59.30	89.85	93.00	87.23	51.30	82.52	66453	8.09
CD-LoRA	**84.80**	85.50	**87.80**	87.30	96.22	87.17	**83.30**	**60.70**	90.22	94.20	**87.34**	50.30	**82.90**	57703	9.29

els. Unless otherwise specified, we adopt the 14B parameter variant with bfloat16 precision.

Consistency Weight λ. To balance the task-specific empirical risk $\mathcal{L}_{\text{task}}$ and the consistency weight $\mathcal{L}_{\text{cons}}$ within the total objective $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda\,\mathcal{L}_{\text{cons}}$, we set $\lambda = 0.05$ across our main experiments.

LoRA Configuration. CD-LoRA applies multiple low-rank adapters to the projection layers (**gate_proj**, **down_proj**, and **up_proj**) of the feed-forward network. We use a LoRA rank of $r = 4$, scaling factor $\alpha = 32$, and a default dropout rate of 0.2. The number of parallel LoRA branches is set to $K = 3$.

Routing and Consistency. When enabled, a lightweight router predicts per-token mixture weights over the K LoRA branches using a softmax-normalized linear projection. CD-LoRA introduces a cross-projection consistency regularization that aligns the latent representations produced by different LoRA branches during training.

Optimization Details. We train for one epoch over the combined multi-task dataset using the AdamW optimizer with a peak learning rate of 2×10^{-4} and a cosine learning rate schedule. The warmup ratio is set to 0.03, and weight decay is disabled. To accommodate large models, we use gradient accumulation with 64 steps and a per-device batch size of 3.

Implementation of Consistency-Driven Alignment During training, we instantiate the consistency-driven alignment objective by performing two stochastic forward passes through the shared LoRA-A projection with independently sampled dropout masks. Let $h^{(1)}$ and $h^{(2)}$ denote the two resulting adapter-space representations. We then compute a symmetric divergence between the two stochastic views and add it to the task loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{cons}}.$$

In our implementation, the adapter-space distributions are modeled with a diagonal Gaussian approximation, which yields an analytic KL divergence and avoids introducing additional trainable parameters. This regularizer is applied only during training and is removed at inference time.

Unlike generic dropout consistency regularization, which is typically applied to output distributions, our alignment is restricted to the shared LoRA-A subspace and is coupled with the de-routed multi-head architecture to target routing-induced training–inference discrepancy in multi-task PEFT.

Data Processing and Sampling All datasets are pre-tokenized using the corresponding Qwen2.5 tokenizer with a maximum sequence length of 512. For computational efficiency, we optionally subsample each task to a fixed number of examples (e.g., 8,000 per task in the 12-task setting). When enabled, datasets are concatenated and randomly shuffled before training to avoid task-order bias.

Reproducibility We fix random seeds for data sampling, model initialization, and training (`seed` $= 7$ unless otherwise stated). All hyper-parameters not explicitly varied in ablation studies are held constant across experiments. Checkpoints are saved every 1,500 steps, and evaluation is performed using the same decoding and metric configurations reported in the main paper.

D Similarity Analysis on 14B Models

Figures 5, 6 and 7 show the layer-wise LoRA-B_i similarity analysis of CD-LoRA, R-LoRA and CR-LoRA on Qwen2.5-14B.

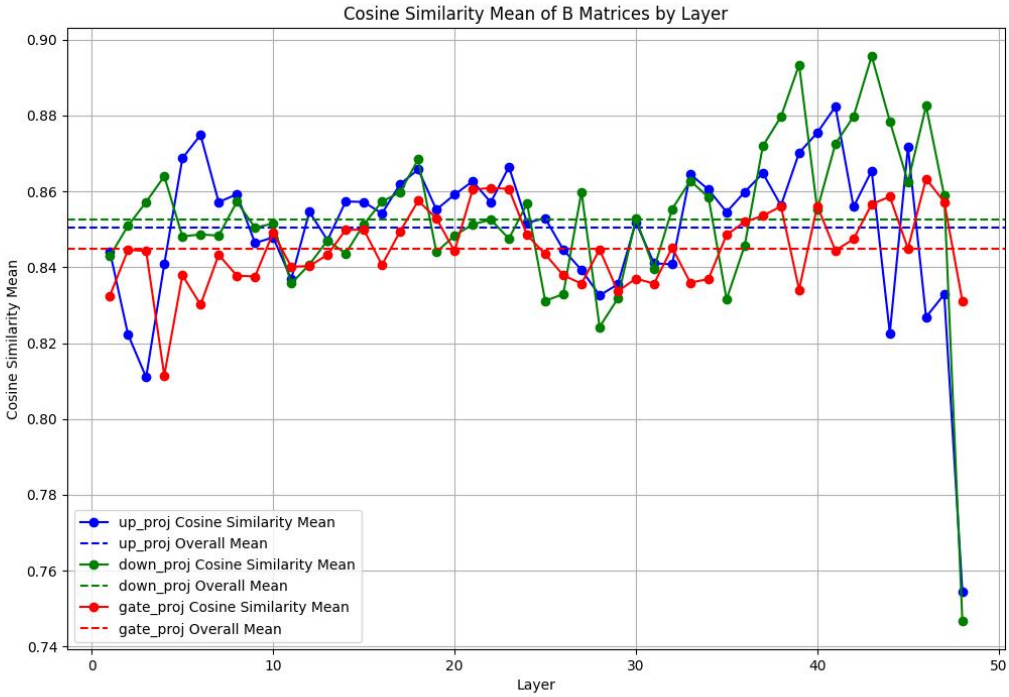

Figure 5: Layer-wise $\mathbf{B}$ matrix similarity heatmap for CD-LoRA. The compact and uniform patterns across all projections validate the efficacy of consistency-driven alignment.

E Sensitivity to the Consistency Weight λ

Figures 8, 9 and 10 report the sensitivity of CD-LoRA to the consistency weight λ on Qwen2.5-3B, Qwen2.5-7B, and Qwen2.5-14B, respectively.

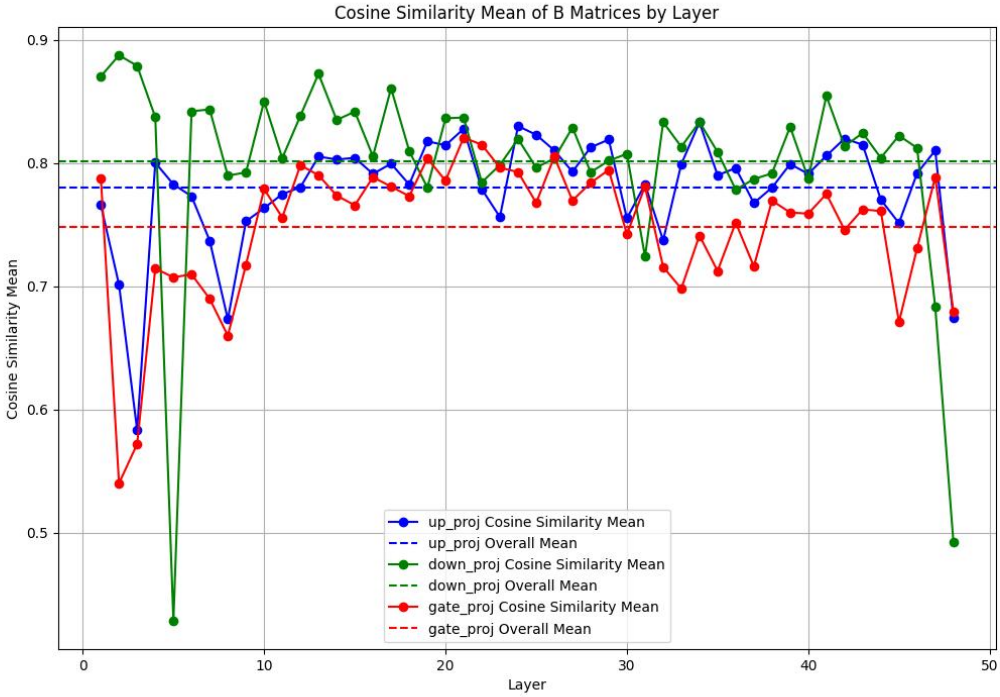

Figure 6: Layer-wise B_i similarity matrix for R-LoRA on Qwen2.5-14B, showing fragmented task-specific subspaces induced by nonlinear routing.

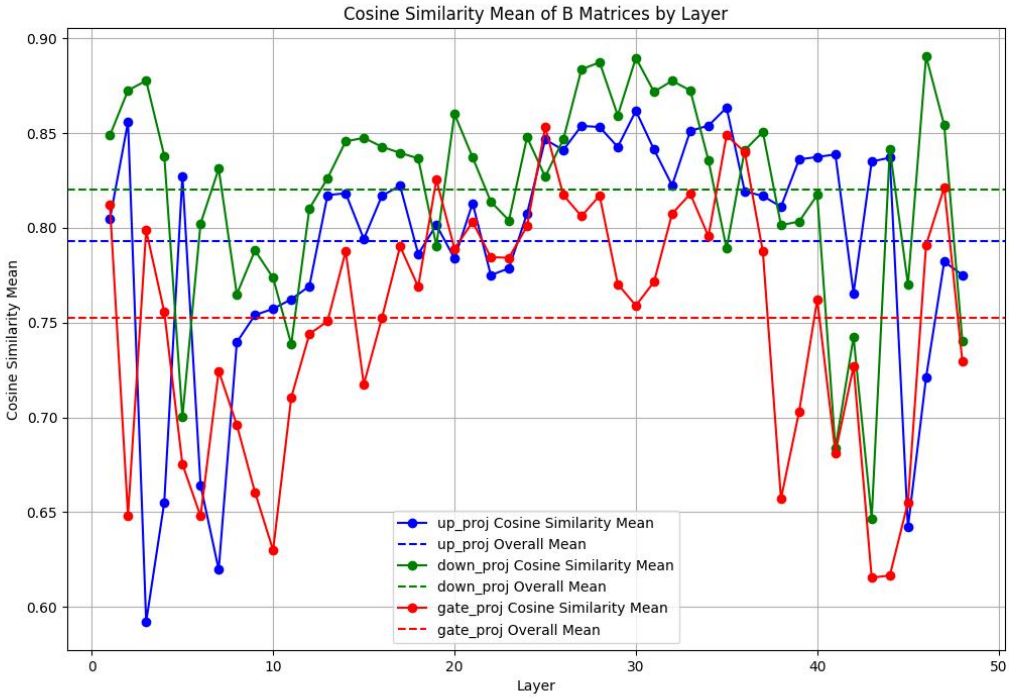

Figure 7: Layer-wise B_i similarity matrix for CR-LoRA. While consistency is enforced, routing still leads to uneven and unstable subspace alignment.

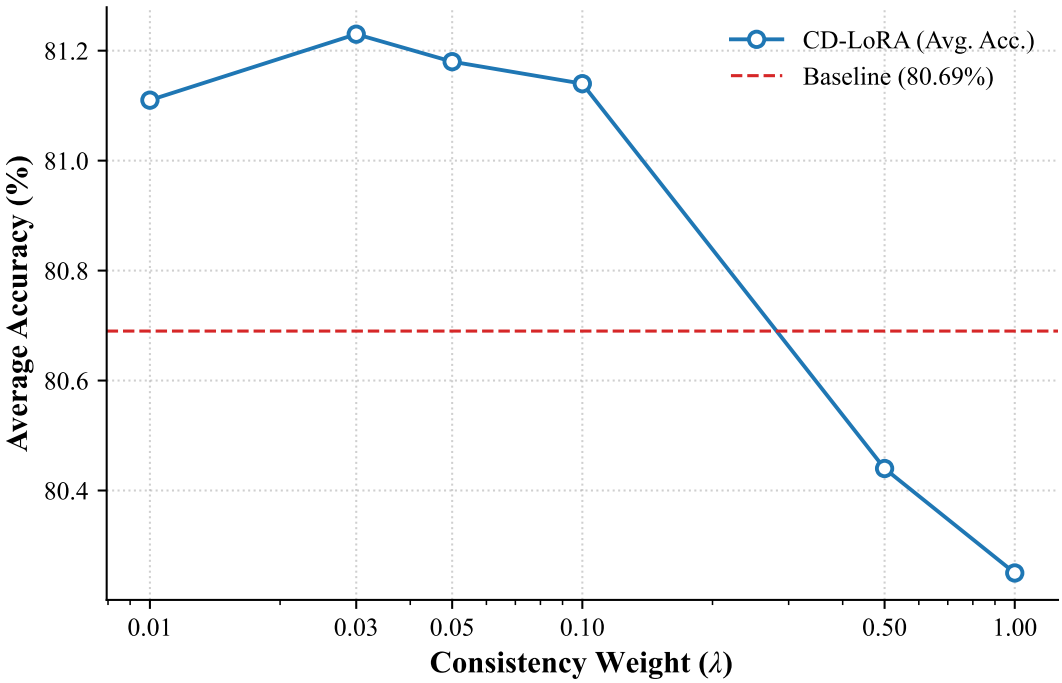

Figure 8: Sensitivity analysis of the consistency-alignment weight λ on Qwen2.5-3B.

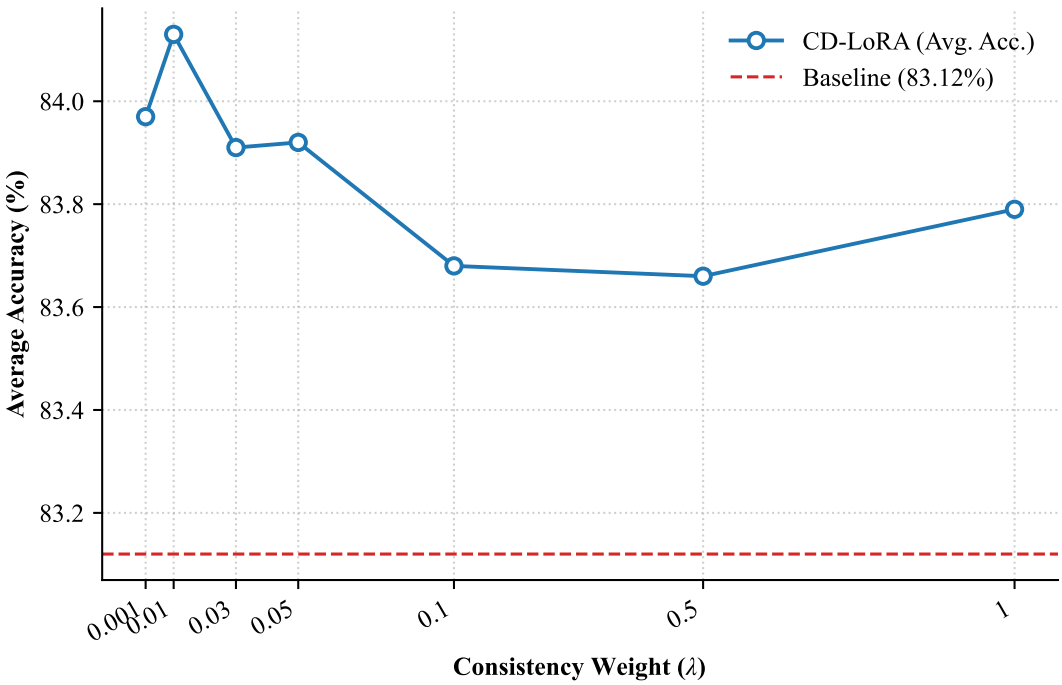

Figure 9: Sensitivity analysis of the consistency-alignment weight λ on Qwen2.5-7B.

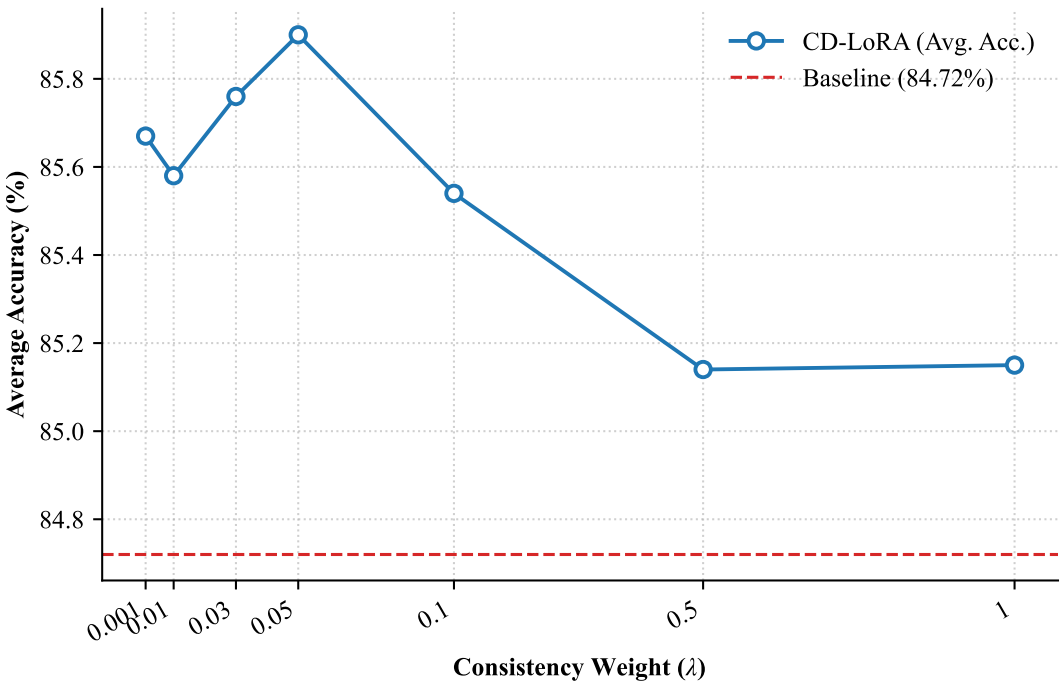

Figure 10: Sensitivity analysis of the consistency-alignment weight λ on Qwen2.5-14B.