From Isolation to Alignment: Unified LoRA for Efficient Multi-Task Learning

Anonymous ACL submission

Abstract

Parameter-Efficient Fine-Tuning (PEFT) is essential for adapting Large Language Models (LLMs) to multi-task scenarios. A prevailing trend in this field involves complex LoRA variants with multiple adapters or heads, which rely on the premise that architectural isolation of task-specific knowledge is necessary. However, this design often introduces dynamic routing, preventing weight merging and causing significant inference latency. In this work, we present a direct challenge to this paradigm. We first reveal a paradox where a simplified, router-free multi-head model with high inter-head redundancy outperforms complex, diversity-driven baselines. Furthermore, we demonstrate that a unified, single-adapter LoRA with increased rank achieves highly competitive performance, questioning the necessity of multi-component structures. Based on these findings, we propose Align-LoRA, a unified and efficient framework that shifts the focus from architectural isolation to representation alignment. Align-LoRA incorporates an explicit alignment loss to encourage the learning of task-shared representations within a shared latent space. Crucially, our method maintains the standard LoRA architecture, ensuring zero inference latency via weight merging. Theoretical analysis and extensive experiments confirm that Align-LoRA significantly surpasses prevailing approaches, establishing a simpler, more effective, and production-friendly paradigm for multi-task PEFT. The code is available at the anonymous link Align-LoRA.

1 Introduction

In recent years, large language models (LLMs) have demonstrated unprecedented performance across a wide range of natural language processing (NLP) tasks (Brown, 2020; Zhao et al., 2026; Chang et al., 2024b). Despite their strong generalization abilities, LLMs often require further adaptation to align with domain-specific requirements or to incorporate updated knowledge (Agiza et al., 2024; Xin et al., 2024). Supervised fine-tuning (SFT) plays a critical role in this process, but full parameter fine-tuning (FFT), which updates all model parameters, poses significant challenges in terms of computational and memory costs (Mao et al., 2025).

To address these demands, parameter-efficient fine-tuning (PEFT) methods have been proposed to adapt LLMs by updating only a small subset of parameters (Han et al., 2024; Chang et al., 2024a). Among these, Low-Rank Adaptation (LoRA) (Hu et al., 2021) has become a widely adopted approach. It approximates the full-rank weight update matrix by decomposing it into two low-rank matrices: a down-projection matrix $\mathbf{A}$ and an up-projection matrix $\mathbf{B}$. In practice, adapting LLMs often involves data from multiple domains or tasks, naturally aligning with the multi-task learning paradigm.

Consequently, this has motivated the development of LoRA variants specifically designed for MTL. An early approach is the Multi-Adapter architecture, which employs multiple, distinct pairs of down-projection ($\mathbf{A}$) and up-projection ($\mathbf{B}$) matrices for different tasks (Wang et al., 2023). To improve parameter efficiency, the Multi-Head architecture was introduced, typically sharing a single $\mathbf{A}$ matrix while maintaining multiple task-specific head matrices ($\mathbf{B}$) (Tian et al., 2024). Furthermore, many of these multi-component architectures employ a routing mechanism, inspired by the Mixture-of-Experts (MoE) framework, to dynamically select or weigh the outputs of different adapters for a given input. Recent prevalent methods like R-LoRA (Liu et al., 2025) further refine this by explicitly encouraging diversity among heads to mitigate redundancy. Figure 1 illustrates these paradigms. **Despite architectural differences, these methods are all built on a common premise: that effective multi-task adaptation requires architectural isolation of task-specific knowledge.**

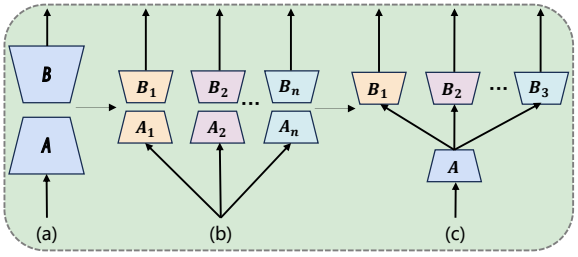

Figure 1: A comparison of LoRA architectural paradigms: (a) Vanilla LoRA; (b) the Multi-Adapter framework; and (c) the Multi-Head framework. The categorization of the multi-component architectures is adapted from R-LoRA (Liu et al., 2025). A common feature of these multi-component designs is the inclusion of a dynamic routing mechanism.

However, architectural isolation has a practical cost: specialized components cannot be merged into the backbone, incurring non-negligible inference latency. This trade-off motivates our work. We first reveal a paradoxical finding: M-LoRA, a simplified model that removes the dynamic router, surpasses its more complex counterparts despite exhibiting higher inter-head similarity. This directly challenges the assumption that component diversity is beneficial, leading to a fundamental question: **Is the multi-component structure truly necessary for effective multi-task adaptation?**

Collectively, these findings point to a new hypothesis: **learning task-shared representations provides a promising alternative to the architectural isolation of task-specific features.** To directly validate this hypothesis and operationalize this principle, we propose Align-LoRA. This method enhances a standard LoRA by augmenting its training objective with a component based on the Kullback-Leibler (KL) Divergence (Kullback and Leibler, 1951), which encourages the alignment of task representations in the shared low-rank space without adding parameters or inference overhead.

Our key contributions are three-fold:

- **Challenging Architectural Isolation:** We demonstrate that simple baselines, specifically a redundant **M-LoRA** and a standard LoRA with **increased rank**, consistently outperform complex multi-component variants. This finding critically questions the necessity of architectural isolation and the additional inference latency it introduces.

- **The Align-LoRA Framework:** We challenge the dominant paradigm that multi-task LoRA requires structural isolation of task-specific knowledge. Instead, we propose and validate that explicitly aligning shared representations in the LoRA latent space is a more effective alternative—realized through **Align-LoRA**, which adds only a representation-level alignment loss to standard LoRA while preserving zero inference latency.

- **Rigorous Validation:** We provide a theoretical analysis of Align-LoRA's effectiveness by deriving a tighter generalization bound. Extensive experiments across diverse models, scales, and task benchmarks validate the practical effectiveness of this principle.

2 Observations

In this section, we critically examine the prevailing assumption that component diversity are essential for effective multi-task adaptation with LoRA. By questioning the fundamental necessity of the prevalent multi-head paradigm, our investigation leads to a new hypothesis centered on the pivotal role of shared knowledge.

2.1 M-LoRA: A Simplified Variant

R-LoRA (Liu et al., 2025) encourages head diversity for task-specific knowledge. To test this hypothesis, we propose **M-LoRA**, an ablation that removes R-LoRA's dynamic router while preserving multi-head randomization and Dropout. M-LoRA aggregates head outputs by simple summation, directly testing whether explicit diversification is necessary. The framework is illustrated in Appendix A.

2.2 The Paradox of Diversity: Less is More

We fine-tune the Qwen2.5-3B (Qwen Team, 2024) model using HydraLoRA, R-LoRA, and M-LoRA on a benchmark comprising five distinct tasks: QNLI (Wang, 2018), PiQA (Bisk et al., 2020), Winogrande (Sakaguchi et al., 2021), ARC (easy & challenge) (Clark et al., 2018), and GSM8K (Cobbe et al., 2021). To quantify inter-head similarity, we compute a matrix of pairwise cosine similarities between all flattened head vectors ($\mathbf{B}_i$). The final metric is the mean of this matrix's off-diagonal values. All Experimental details in this work, including implementation specifics, hyperparameter settings, dataset descriptions, baseline configurations, and other relevant information, are documented in the Appendix H.

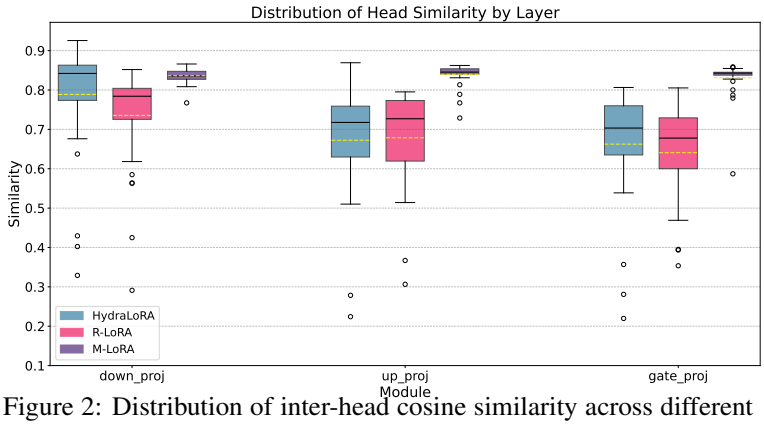

Figure 2: Distribution of inter-head cosine similarity across different model modules. Within each module, the box plots from left to right correspond to HydraLoRA, R-LoRA, and M-LoRA, respectively. The solid line within each box indicates the median similarity, while the dashed line represents the mean.

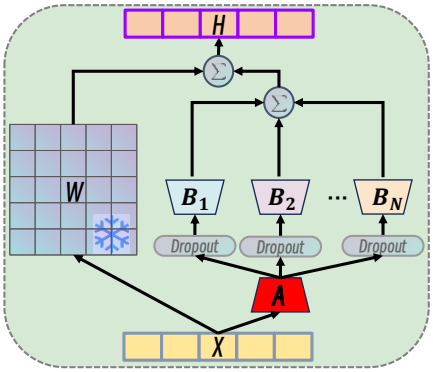

Figure 3: The framework of M-LoRA. Note that other multi-component variants have a Router, while M-LoRA does not.

Table 1: Comparative study of several multi-head LoRA variants across five tasks.

Schemes	QNLI	PiQA	Winogrande	ARC	GSM8K	Avg	%Para
HydraLoRA	81.91	84.21	70.92	87.21	45.95	74.04	0.45
w/o Router	81.33	83.51	70.14	86.46	45.35	73.58	0.41
R-LoRA	82.03	85.55	71.84	87.69	46.25	74.67	0.45
M-LoRA	**82.52**	**86.76**	**72.95**	**88.15**	**46.85**	**75.45**	**0.41**

Our findings reveal a paradox regarding the role of head diversity in multi-task adaptation. Figure 2, which plots the inter-head cosine similarity, shows that R-LoRA successfully achieves its design goal of maximizing diversity, exhibiting the lowest similarity. In stark contrast, M-LoRA, which lacks any diversity-enforcing mechanism, displays the opposite effect, yielding a high degree of head redundancy with similarity medians consistently exceeding 0.85. Paradoxically, as demonstrated in Table 1, this high-redundancy model achieves superior multi-task performance. Despite its architectural simplicity, M-LoRA consistently and significantly outperforms the more complex HydraLoRA and R-LoRA. This outcome presents a fundamental contradiction to the philosophy of prior work: the architectural configuration that seemingly violates the principle of head diversity actually enhances multi-task generalization.

A natural concern is that this advantage may stem from head count coinciding with task count (N=5), implicitly allowing each head to specialize. To test this, we vary N while keeping five tasks fixed (see Appendix G.2). Even when heads must be shared across tasks (N=2 or N=3), M-LoRA remains competitive (74.14 and 74.62 vs. 75.45 at N=5), confirming that the router-free design handles task sharing without explicit routing mechanisms. The slight advantage at N=5 reflects the natural correspondence between heads and tasks, not a dependency on this configuration.

Extended training (1–3 epochs) in Appendix G.1 shows that while R-LoRA's router theoretically subsumes M-LoRA, the additional parameters create a harder optimization landscape; M-LoRA's advantage not only persists but widens ($+0.78 \rightarrow +1.23$pp, see Appendix G.1), ruling out convergence artifacts.

2.3 From Observation to Hypothesis

This section explains M-LoRA's effectiveness. Its strong multi-task generalization challenges conventional assumptions and offers a new perspective on multi-task adaptation in LoRA. Improving multi-task learning has followed two distinct paths: isolating **task-specific** knowledge to mitigate interference, or enhancing **task-shared** knowledge to improve generalization. While recent methods like LoRA MoE (Dou et al., 2023) and R-LoRA (Liu et al., 2025) focus on isolating task-specific knowledge, **the alternative path of actively enhancing task-shared knowledge within the LoRA framework remains unexplored.**

Table 2: Comparison of different training schemes on LLaMA2. All variants use a LoRA rank of 8, except LoRA†(rank=30, adjusted to match multi-head variants' trainable parameter count). * indicates results from (Tian et al., 2024).

Metrics	Base	LoRA	LoRAHub*	LoRA MoE*	HydraLoRA	R-LoRA	LoRA†	M-LoRA
7B	31.61	37.05	39.70	40.30	41.46	42.24	42.21	**42.83**
13B	38.42	40.73	41.90	43.70	44.31	44.96	45.02	**46.16**
% Param	-	0.06	1.24	2.98	0.34	0.34	0.34	0.32

Table 3: Comparison of different training schemes on Qwen2.5. The superscript in "LoRA" (e.g., ^{4}, ^{8}, etc.) indicates the rank value used for each variant.

Metrics	Base	LoRA^{4}	LoRA^{8}	LoRA^{9}	LoRA^{10}	HydraLoRA	R-LoRA	M-LoRA
7B	39.82	43.21	46.66	48.18	49.51	49.12	49.51	**49.74**
14B	45.33	48.18	51.82	52.74	**54.23**	53.76	54.08	54.18
Rank	-	4	8	9	10	4	4	4
% Param	-	0.10	0.20	0.22	0.25	0.25	0.25	0.22

M-LoRA challenges this specialization-focused paradigm. We hypothesize that high similarity is not a sign of failed specialization, but a feature derived from the architecture's implicit regularization. The key mechanism is the interplay between removing the router and retaining multi-head dropout. In models like R-LoRA, heads act as competing "specialists" via a dynamic router, which can lead to redundancy or load imbalance. In contrast, by replacing the router with simple averaging, M-LoRA compels the multiple $\mathbf{B}$ heads to form a collaborative ensemble. As illustrated in Figure 3, retaining the multi-head dropout forces each head to learn from different input perspectives. By requiring all heads to contribute via summation, they are compelled to converge on a robust, task-general representation.

To validate this mechanism, we performed an ablation on a non-dropout multi-head variant, HydraLoRA. As shown in Table 1, removing the router from HydraLoRA ('w/o Router') causes its performance to drop, while M-LoRA's collaborative ensemble achieves the highest performance. To provide a rigorous causal proof, we conduct a full 2×2 factorial analysis (Router $\times$ Dropout) detailed in Appendix G.3. The empirical results show that removing dropout leads to a severe performance degradation (-1.63pp), an effect over $3\times$ larger than that of router removal (-0.46pp). This quantitative attribution firmly establishes that dropout-based collaborative ensembling—rather than the mere absence of routing—is the primary driver of M-LoRA's advantage, transforming isolated "specialists" into effective "collaborators" and significantly enhancing task-general learning.

The success of M-LoRA suggests a revised viewpoint: **learning task-shared representations provides a promising path towards multi-task learning, offering a powerful alternative to the architectural isolation of task-specific features.**

3 Increasing Rank: A Unified Adapter

Given M-LoRA's strong performance with highly redundant heads, which largely learn shared knowledge, this section explores a critical question: Is the multi-head architecture itself truly necessary for multi-task generalization, or does it merely serve as a means to increase total trainable parameters rather than offering genuine benefits?

To test this, we design a straightforward yet powerful experiment. We abandon the multi-component structure entirely and instead use a standard, single-adapter LoRA. We reallocate the entire parameter budget of the complex variants into this single adapter by simply increasing its rank, r. Following the experimental setup of HydraLoRA (Tian et al., 2024), we conduct fine-tuning on a curated subset of the Flanv2 dataset (Liu et al., 2022). This training data is sampled from dozens of individual datasets and organized into ten distinct task categories, providing comprehensive training across both NLU and NLG capabilities. We then evaluate the models' multi-task generalization on the challenging BBH(3-shot) benchmark (Suzgun et al., 2023), which is designed to test generalization.

The results, presented in Table 2 and Table 3, reveal a clear trend. Across different base models, the performance of a standard LoRA adapter consistently improves with its rank. Crucially,

when its rank is scaled to a comparable parameter count, a simple, single-adapter LoRA achieves performance that is competitive with sophisticated multi-component architectures such as LoRA-Hub (Huang et al., 2023), LoRA MoE (Liu et al., 2024a), HydraLoRA (Tian et al., 2024), and R-LoRA (Liu et al., 2025).

This finding provides compelling evidence that **the architectural complexity of multi-adapter designs may not be a prerequisite for achieving strong multi-task generalization.** Our results indicate that a simple, unified adapter with sufficient capacity delivers comparable performance. This challenges not only the trend toward elaborate structures but also the underlying strategy of isolating task-specific features, suggesting it is a less effective path to generalization than previously assumed and that the research focus on specialized components may warrant reconsideration.

4 Beyond Rank: Align-LoRA

Our investigation in the preceding sections has led to two critical conclusions. First, based on our analysis in Section 2, we formed a guiding hypothesis: learning task-general, shared knowledge may be more critical than enforcing task-specific separation. Second, our findings demonstrate that the architectural complexity introduced by multi-component designs is unnecessary for achieving strong multi-task generalization. This calls into question the prevailing assumption that specialized structures are a prerequisite for effective multi-task learning.

These conclusions motivate a shift from structural isolation toward a more principled approach. By adopting the standard, high-rank LoRA as an efficient baseline, we move beyond merely increasing rank to address a fundamental challenge: **how to explicitly validate and enhance the learning of shared representations within a unified adapter.** To this end, we introduce **Align-LoRA**, a novel framework that bridges our theoretical hypothesis with a concrete alignment mechanism.

4.1 Methodology

To enhance multi-task generalization, we introduce **Align-LoRA**, a method that encourages the model to learn task-shared features. Align-LoRA introduces an alignment loss, $\mathcal{L}_{\text{align}}$, to explicitly minimize the statistical distance between the low-dimensional representations generated by the shared LoRA down-projection matrix, $\mathbf{A}$. **To the best of our knowledge, this is the first work to systematically apply statistical distance metrics for this purpose within the multi-task LoRA framework**, drawing inspiration from their foundational use in domain adaptation (Pan et al., 2010).

Our primary approach instantiates $\mathcal{L}_{\text{align}}$ using the **Kullback-Leibler (KL) divergence** (Kullback and Leibler, 1951), a classic statistical method for measuring the distance between distributions. However, we hypothesize that the principle of aligning representations is broadly applicable and not contingent on a single metric. To validate this, we introduce a variant that employs the multi-kernel **Maximum Mean Discrepancy (MK-MMD)** (Gretton et al., 2012). The strong performance of both instantiations, as we will demonstrate, validates our core thesis: that explicitly aligning the low-dimensional representations of different tasks is a robust and viable strategy. The formulation for MK-MMD is detailed in Appendix E.

Let $\mathcal{T} = \{T_1, T_2, \ldots, T_M\}$ be a set of M tasks. For an input $\mathbf{x}$ from task T_i with contextualized embeddings X_{T_i}, the representation we align is the output of the down-projection matrix:

$$\phi_{T_i}(\mathbf{x}) = \mathbf{A} \cdot X_{T_i}. \tag{1}$$

Our choice to operate on this rank-r latent space is motivated by two key factors. First, it directly targets the component responsible for shared knowledge. Recent studies have consistently found that the down-projection matrix $\mathbf{A}$ tends to learn task-general features while the up-projection matrix $\mathbf{B}$ captures task-specific knowledge (Agiza et al., 2024; Wang et al., 2024; Tian et al., 2024). By applying our alignment loss to the output of $\mathbf{A}$, we directly enhance its natural function, promoting the development of robust, shared representations. Second, this approach is highly efficient. As demonstrated in prior work (Liu et al., 2025), performing operations in the low-dimensional space significantly reduces computational load and GPU memory demand, ensuring our method's practicality.

To measure and minimize the distance between the representation distributions from different tasks, we model the batch-wise distribution for each task T_i as a multivariate Gaussian with a diagonal covariance matrix, $\mathcal{N}(\boldsymbol{\mu}_i, \text{diag}(\boldsymbol{\sigma}_i^2))$ (justified in Appendix D). The mean $\boldsymbol{\mu}_i$ and variance $\boldsymbol{\sigma}_i^2$ are empirically estimated from the output vectors $\{\phi_{T_i}(\mathbf{x})\}$

Table 4: Multi-task generalization performance on the BBH benchmark. Align-LoRA (A-LoRA), particularly A-LoRA-K, demonstrates a clear advantage over the other variants across model families and scales.

Metrics	LoRA	LoRAMoE	HydraLoRA	R-LoRA	M-LoRA	A-LoRA-M	A-LoRA-K
Qwen2.5-7B	48.36	47.18	47.38	48.32	48.44	47.53	**50.28**
LLaMA3-8B	44.89	44.18	44.03	45.01	45.35	45.42	**48.84**
Qwen2.5-14B	52.93	50.74	51.92	52.21	53.78	52.24	**55.11**
Rank	10	4	4	4	4	8	8
% Param	0.25	0.38	0.25	0.25	0.22	**0.20**	**0.20**

Table 5: Performance comparison on an 8-task multi-task reasoning benchmark, evaluated on the Qwen2.5-3B and Qwen2.5-7B models. Baselines include the original LoRA and several multi-component variants. Our proposed Align-LoRA (A-LoRA) consistently achieves significantly superior performance, demonstrating its strong multi-task generalization capabilities.

Schemes	Task1	2	3	4	5	6	7	8	Avg	%Par
Qwen2.5-3B										
LoRA	86.31	56.42	84.65	72.76	91.37	87.91	87.60	44.80	76.48	0.45
LoRAMoE	87.41	58.21	85.64	73.37	92.18	87.40	87.35	44.80	77.05	0.68
HydraLoRA	86.58	56.42	85.00	73.36	92.18	87.33	88.38	45.15	76.80	0.45
R-LoRA	87.12	57.95	88.13	73.89	94.71	88.25	88.26	45.60	77.99	0.45
M-LoRA	88.02	57.95	88.87	74.21	94.71	88.91	89.07	46.35	78.51	0.42
A-LoRA-M	87.94	58.03	88.87	74.12	94.51	88.85	88.61	45.88	78.35	0.42
A-LoRA-K	**89.25**	**59.88**	**90.35**	**75.41**	**95.33**	**89.55**	**91.95**	**48.75**	**80.06**	**0.42**
Qwen2.5-7B										
LoRA	88.41	60.78	88.42	81.58	93.52	91.20	91.79	48.15	80.48	0.25
LoRAMoE	89.52	61.44	88.86	82.94	92.87	91.54	91.89	48.72	80.97	0.38
HydraLoRA	88.66	61.23	89.55	81.72	93.57	91.67	91.74	48.70	80.86	0.25
R-LoRA	89.80	62.51	89.36	83.78	95.12	91.02	92.17	50.15	81.74	0.25
M-LoRA	91.35	62.51	91.98	84.70	95.93	91.02	91.97	50.20	82.46	0.22
A-LoRA-M	90.86	62.45	91.68	84.59	95.93	90.74	91.75	50.45	82.31	0.20
A-LoRA-K	**92.23**	**64.85**	**92.89**	**85.73**	**95.93**	**93.35**	**92.93**	**53.66**	**83.95**	**0.20**

in a given batch. Since standard KL divergence is asymmetric, we employ a symmetric formulation. The total alignment loss, $\mathcal{L}_{\text{KL}}$, is the sum of these symmetric pairwise divergences across all unique task pairs (T_i, T_j) where $i < j$:

$$\mathcal{L}_{\text{KL}} = \sum_{i=1}^{M} \sum_{j=i+1}^{M} \frac{1}{2} \left(D_{\text{KL}}(p_{T_i} \| p_{T_j}) + D_{\text{KL}}(p_{T_j} \| p_{T_i}) \right), \tag{2}$$

where p_{T_i} is the modeled Gaussian distribution over the low-dimensional representations of task T_i. This loss drives the empirical mean and variance of each task's distribution toward a common value.

The alignment loss is incorporated as an auxiliary objective to the primary language modeling task. The total loss function is therefore defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{lm}} + \lambda \cdot \mathcal{L}_{\text{align}}, \tag{3}$$

where $\mathcal{L}_{\text{lm}}$ is the primary language modeling loss, $\mathcal{L}_{\text{align}}$ is the auxiliary alignment loss ($\mathcal{L}_{\text{KL}}$), and λ is a scalar hyperparameter controlling the influence of the auxiliary task.

A key advantage of Align-LoRA is its compatibility with various LoRA-based strategies and initialization schemes. Importantly, unlike multi-component LoRA variants, Align-LoRA introduces no additional modules that increase overhead. Consequently, its trained weights can be merged directly into the base model, incurring **zero inference latency**. This property ensures both efficiency and practicality, making Align-LoRA a lightweight yet effective solution for multi-task adaptation. For a more detailed analysis of this inference efficiency, please refer to Appendix B.

4.2 Experiment

We evaluate **Align-LoRA** (A-LoRA) against standard LoRA and multi-component variants. Two alignment approaches are considered: **A-LoRA-K** (KL divergence) and **A-LoRA-M** (MMD). De-

Table 6: Comparison of training times and related metrics for different methods.

Metric	LoRA	HydraLoRA	Align-LoRA
Rank	10	4	8
Epoch Time	3.13h	3.53h	**3.03h**
%Param	0.45	0.45	**0.42**
FLOPs 10^{18}	3.545	3.986	**3.539**
Performance	76.48	76.80	**80.06**

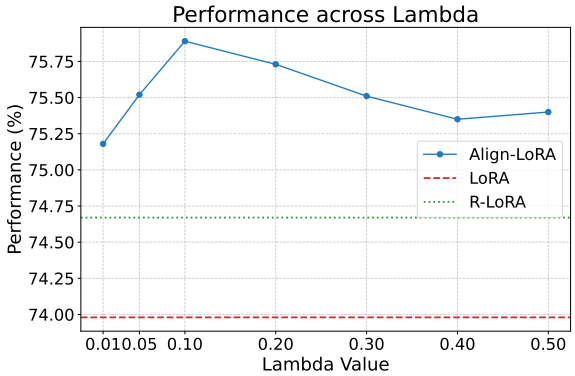

Figure 4: Effect of Hyperparameter λ on Performance

tailed experimental setup, dataset descriptions, and baseline configurations are provided in Appendix H.3 and Appendix J.

Multi-task generalization. We fine-tune on the five-task benchmark (Section 2) and evaluate on unseen BBH tasks. Table 4 shows that A-LoRA-K outperforms all baselines across model families (Qwen2.5 (Qwen Team, 2024), LLaMA3 (for the ExoMars Rover Mission to Mars, 2024)) and scales (7B–14B), with fewer trainable parameters.

Multi-task adaptation. On an eight-task in-domain benchmark (Table 5), A-LoRA-K achieves the best average score across 3B and 7B models, again using the lowest parameter budget. Statistical significance is confirmed by three independent runs ($p < 0.001$, see Appendix G.4).

More Result. Beyond the main benchmarks, A-LoRA-K exhibits consistent robustness across a wide range of λ values without delicate tuning (Figure 4) and maintains its advantage on highly heterogeneous task sets spanning translation, struct-to-text, and diverse reasoning domains (Appendix G.6). The method further proves structurally universal: it improves performance when applied to either Attention or MLP modules (Appendix G.5) and integrates seamlessly into existing multi-head architectures such as R-LoRA and M-LoRA, yielding additional gains (Appendix G.7). Training efficiency is competitive: Align-LoRA achieves the lowest FLOPs and per-epoch time among all compared methods (Table 6). t-SNE visualizations confirm that alignment brings task representations closer while keeping task clusters clearly distinguishable, ruling out concerns of feature collapse (Appendix I).

Overall, these results show that explicit representation alignment in the shared LoRA latent space consistently improves multi-task generalization over both standard LoRA and complex multi-component variants, without sacrificing mergeability or inference efficiency.

4.3 Theoretical Analysis

To theoretically analyze Align-LoRA's generalization performance in multi-task scenarios, we derive a novel generalization bound based on statistical learning theory. Our key insight is that explicitly aligning cross-task representation distributions effectively reduces the generalization gap, yielding a significantly tighter generalization bound compared to traditional multi-component LoRA variants. For the complete derivation, technical assumptions, and proofs, please refer to Appendix F.

Formally, let M be the number of tasks, with $\mathcal{D}_i$ and $\hat{\mathcal{D}}_i$ denoting the data distribution and training dataset for task i, respectively. Let $\hat{R}_i(f)$ be the empirical training risk on $\hat{\mathcal{D}}_i$, and n_{total} be the total sample size across all tasks. The generalization bound for Align-LoRA is given by:

$$R_{\text{MTL}}(f) \leq \frac{1}{M}\sum_{i=1}^{M}\hat{R}_i(f) + \frac{\lambda}{M}\sum_{i<j}\Delta(\mathcal{D}_i, \mathcal{D}_j) + \mathcal{O}\left(\sqrt{\frac{M\log(M/\delta)}{n_{\text{total}}}}\right), \quad (4)$$

where $R_{\text{MTL}}(f)$ is the average expected risk, and $\Delta(\mathcal{D}_i, \mathcal{D}_j)$ measures the pairwise distribution discrepancy. Here, λ is a weighting hyperparameter that subsumes theoretical constants related to the model's Lipschitz properties.

The bound highlights the mechanism of Align-LoRA: by actively minimizing the pairwise discrepancy $\Delta(\mathcal{D}_i, \mathcal{D}_j)$ during training, Align-LoRA significantly reduces the second term (the alignment term). While this constraint imposes a regularization on the representation space, our analysis suggests that for over-parameterized models like LLMs, the substantial reduction in the distribution

discrepancy dominates, leading to a tighter overall generalization gap and superior test performance.

5 Related Works

5.1 Low-Rank Adaptation (LoRA)

LoRA (Hu et al., 2021) adapts Large Language Models (LLMs) by decomposing the weight update $\Delta\mathbf{W}$ of a dense matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$ into two low-rank matrices, $\mathbf{A} \in \mathbb{R}^{r\times n}$ and $\mathbf{B} \in \mathbb{R}^{m\times r}$, assuming a low intrinsic dimensionality. Choosing a rank $r \ll \min(m, n)$ reduces trainable parameters from $\mathcal{O}(mn)$ to $\mathcal{O}(r(m + n))$. The modified forward pass is formulated as:

$$h = \mathbf{W}x + \Delta\mathbf{W}x = \mathbf{W}x + \mathbf{B}\mathbf{A}x. \quad (5)$$

Post-training, the low-rank update $\Delta\mathbf{W} = \mathbf{B}\mathbf{A}$ can be merged back into the original weights $\mathbf{W}$, ensuring zero inference overhead.

Subsequent advancements build on this foundation to optimize efficiency and performance: AdaLoRA (Zhang et al., 2023) dynamically allocates the rank budget, DoRA (Liu et al., 2024b) decouples updates into magnitude and direction, while PiSSA (Meng et al., 2025) and NLoRA (Guo et al., 2025) introduce advanced initialization and decomposition strategies.

5.2 Multi-Component LoRA

To adapt LoRA for multi-task learning, a natural extension is to employ multiple trainable components. Early works proposed the **Multi-Adapter** architecture, which utilizes multiple independent LoRA adapters (i.e., distinct $\mathbf{B}_i\mathbf{A}_i$ pairs) for different tasks. Notable examples of this approach include Multi-LoRA (Wang et al., 2023), MixLoRA (Li et al., 2024), LoRAMoE (Dou et al., 2023), MoELoRA (Liu et al., 2024a), and LoRAHub (Huang et al., 2023).

The **Multi-Head** architecture improves parameter efficiency by exploiting the distinct roles of LoRA's matrices: the down-projection ($\mathbf{A}$) captures shared, task-general features, while multiple up-projection heads ($\mathbf{B}_i$) learn task-specific features. Methods like HydraLoRA (Tian et al., 2024), MALoRA (Wang et al., 2024), MTLLoRA (Agiza et al., 2024), and R-LoRA (Liu et al., 2025) employ a shared $\mathbf{A}$ with multiple $\mathbf{B}_i$ heads. R-LoRA further introduces random initialization to reduce head similarity. Figure 1 illustrates these paradigms. The aggregated weight update is a dynamically weighted sum of each head's output:

$$\Delta\mathbf{W} = \sum_{i=1}^{N} \omega_i(\mathbf{x}) \cdot \mathbf{B}_i\mathbf{A}. \quad (6)$$

where $\omega_i(\mathbf{x})$ represents input-dependent mixing weights. Drawing inspiration from the Mixture-of-Experts (MoE) framework, these weights are determined by a dynamic routing mechanism (e.g., a softmax gating network) that adaptively assigns importance scores to each "expert" head based on the input $\mathbf{x}$.

However, this complexity introduces a critical trade-off. A significant drawback of input-dependent routing is that the aggregated update $\Delta\mathbf{W}$ can no longer be pre-computed. Consequently, the adapter weights **cannot be merged** into the frozen backbone model post-training. This results in **non-negligible inference latency**, as the router and multiple heads must be processed for each forward pass, sacrificing one of LoRA's most significant practical advantages.

6 Conclusion

This work challenges the assumption that multi-task LoRA adaptation requires architectural isolation. Through controlled experiments, we show that simpler alternatives, such as a router-free multi-head ensemble (M-LoRA) and a higher-rank single adapter, match or outperform complex routed designs. Building on this insight, we introduce **Align-LoRA**, which aligns cross-task representations in the shared LoRA latent space, achieving strong multi-task generalization with zero inference latency.

A theoretical analysis further shows that reducing cross-task representation discrepancy tightens the multi-task generalization bound, providing qualitative support for the alignment principle. More broadly, our findings suggest that for multi-task PEFT, representation alignment in a shared latent space is a practical alternative to structural decomposition. The mergeable, single-adapter design makes the approach straightforward to deploy. Future work can explore alignment objectives that scale sub-quadratically with task count, and test whether alignment followed by selective specialization combines the advantages of both paradigms.

Limitations

Although the proposed framework demonstrates strong empirical performance, this study has certain limitations. Due to constraints in computational resources, our evaluations are restricted to mainstream model scales ranging from 3B to 14B parameters, and the scalability of this approach on ultra-large foundation models with hundreds of billions of parameters remains to be verified. Additionally, our experiments are conducted entirely on open-source benchmarks, meaning the method has not yet been validated through deployment in actual production environments with real-world user traffic or proprietary data distributions.

References

Ahmed Agiza, Marina Neseem, and Sherief Reda. 2024. Mtlora: Low-rank adaptation approach for efficient multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16196–16205.

Peter L Bartlett and Shahar Mendelson. 2002. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of machine learning research*, 3(Nov):463–482.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19.

Yonatan Bisk, Rowan Zellers, Ronan Le bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Yupeng Chang, Yi Chang, and Yuan Wu. 2024a. Balora: Bias-alleviating low-rank adaptation to mitigate catastrophic inheritance in large language models. *arXiv preprint arXiv:2408.04556*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Weirong Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qian Yang, and Xingxu Xie. 2024b. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Xilun Chen and Claire Cardie. 2018. Multinomial adversarial networks for multi-domain text classification. *arXiv preprint arXiv:1802.05694*.

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. In *NAACL*.

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:1803.05457v1*.

Karl Cobbe, Vineet Kosaraju, Mo 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. *arXiv preprint arXiv:2110.14168*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 4(7).

The Raman Laser Spectrometer for the ExoMars Rover Mission to Mars. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K Sriperumbudur. 2012. Optimal kernel choice for large-scale two-sample tests. *Advances in neural information processing systems*, 25.

Chenlu Guo, Yuan Wu, and Yi Chang. 2025. Nlora: Nyström-initiated low-rank adaptation for large language models. *Preprint*, arXiv:2502.14482.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*.

Dou Hu, Lingwei Wei, Wei Zhou, and Songlin Hu. 2025. Impartial multi-task representation learning via variance-invariant probabilistic decoding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19883–19897, Vienna, Austria. Association for Computational Linguistics.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

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*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

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. *arXiv preprint arXiv:2404.15159*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

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

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. 2024a. When moe meets llms: Parameter efficient fine-tuning for multi-task medical applications. *Preprint*, arXiv:2310.18339.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024b. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015a. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015b. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Yuren Mao, Yuhang Ge, Yijiang Fan, Wenyi Xu, Yu Mi, Zhonghao Hu, and Yunjun Gao. 2025. A survey on lora of large language models. *Frontiers of Computer Science*, 19(7):197605.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2025. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of machine learning*. MIT press.

Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. 2010. Domain adaptation via transfer component analysis. *IEEE transactions on neural networks*, 22(2):199–210.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

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 LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Dino Sejdinovic, Bharath Sriperumbudur, Arthur Gretton, and Kenji Fukumizu. 2013. Equivalence of distance-based and rkhs-based statistics in hypothesis testing. *The annals of statistics*, pages 2263–2291.

Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

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.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. 2024. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *Preprint*, arXiv:2404.19245.

Alex Wang. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Xujia Wang, Haiyan Zhao, Shuo Wang, Hanqing Wang, and Zhiyuan Liu. 2024. Malora: Mixture of asymmetric low-rank adaptation for enhanced multi-task learning. *arXiv preprint arXiv:2410.22782*.

Yiming Wang, Yu Lin, Xiaodong Zeng, and Guannan Zhang. 2023. Multilora: Democratizing lora for better multi-task learning. *arXiv preprint arXiv:2311.11501*.

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

Yuan Wu. 2024. Margin discrepancy-based adversarial training for multi-domain text classification. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 170–182. Springer.

Chunlei Xin, Yaojie Lu, Hongyu Lin, Shuheng Zhou, Huijia Zhu, Weiqiang Wang, Zhongyi Liu, Xianpei Han, and Le Sun. 2024. Beyond full fine-tuning: Harnessing the power of lora for multi-task instruction tuning. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2307–2317.

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*.

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*.

Chengshuai Zhao, Zhen Tan, Pingchuan Ma, Dawei Li, Bohan Jiang, Yancheng Wang, Yingzhen Yang, and Huan Liu. 2025. Is chain-of-thought reasoning of llms a mirage? a data distribution lens. *arXiv preprint arXiv:2508.01191*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, and 3 others. 2026. A survey of large language models. *Preprint*, arXiv:2303.18223.

A M-LoRA

Figure 3 illustrates the framework of M-LoRA, a variant of R-LoRA. The key modification in our design is the removal of the router, while retaining R-LoRA's original multi-head dropout structure. Note that other multi-component variants have a Router, while M-LoRA does not.

B Inference Efficiency of Align-LoRA

In this section, we explain why single-adapter architectures such as Align-LoRA avoid introducing extra inference overhead. Methods that maintain a single-adapter architecture, such as our proposed **Align-LoRA**, retain a crucial practical advantage: the ability to merge adapter weights with the backbone model, resulting in **zero inference overhead**. For a pre-trained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d\times k}$, a single low-rank update $\Delta\mathbf{W} = \mathbf{BA}$ is learned. Before deployment, this update can be seamlessly integrated into the original weights:

$$\mathbf{W}' = \mathbf{W}_0 + \mathbf{BA}. \tag{7}$$

Inference then proceeds using the modified weights, $h = \mathbf{W}'x$, without requiring any additional parameters or computational steps.

In sharp contrast, multi-component architectures like HydraLoRA forfeit this efficiency. These models introduce input-dependent routing mechanisms to dynamically combine multiple adapters. For an input x, a router first computes weights $\omega = \text{softmax}(\mathbf{W}_r x)$, which are then used to create a blended output:

$$h = \left(\mathbf{W}_0 + \sum_{i=1}^{N} \omega_i \mathbf{B}_i \mathbf{A}_i\right) x. \tag{8}$$

Because the routing weights ω are input-dependent, the adapter matrices $(\mathbf{B}_i, \mathbf{A}_i)$ cannot be pre-merged into $\mathbf{W}_0$. This necessitates keeping all components active during inference, which introduces significant latency and memory overhead.

Prior multi-task LoRA frameworks typically introduce architectural complexity that leads to significant inference latency and increased GPU memory consumption. In contrast, Align-LoRA is designed to achieve effective multi-task learning while preserving the original's advantage of **zero computational overhead**.

C Training Efficiency of Align-LoRA

Table 6 presents the training efficiency metrics across different methods. The results highlight the superior efficiency of Align-LoRA, which achieves the **lowest computational cost (FLOPs)** and the **shortest training time**, outperforming both the standard LoRA baseline and the multi-head HydraLoRA. Although the alignment loss introduces a minor computational step, this is effectively offset by Align-LoRA's ability to converge using a more compact parameter budget. Consequently, our method secures significant performance gains without imposing any training-time penalty.

D Gaussian Approximation

The diagonal-Gaussian assumption used in the KL alignment objective is justified on three grounds:

1. **Central Limit Theorem:** The representation $\phi(\mathbf{x}) = \mathbf{A} \cdot X$ is a linear projection from a high-dimensional embedding space ($\geq$2048 dimensions) to a low-rank latent space (r=8). By the Central Limit Theorem, the batch-wise statistics of such projections converge rapidly

to the Gaussian at typical training batch sizes ($\geq$128 samples per task).

2. **Diagonal structure as beneficial regularization:** Modeling the covariance as diagonal prevents overfitting to spurious cross-dimensional correlations from limited per-task batch statistics. This is analogous to the well-known observation in variational autoencoders that diagonal-covariance models often outperform full-covariance variants in practice.

3. **Non-parametric validation:** The MMD variant (A-LoRA-M) makes no distributional assumptions and still outperforms baselines on most benchmarks (Table 5), confirming that the alignment principle does not depend on the Gaussian approximation. The stronger performance of the KL variant under this approximation is a beneficial outcome of the structured inductive bias, not a fragility of the method.

E Alignment with Maximum Mean Discrepancy (MMD)

As a complementary, non-parametric approach, we also investigate the Maximum Mean Discrepancy (MMD) (Sejdinovic et al., 2013), specifically its robust multi-kernel extension, MK-MMD (Gretton et al., 2012). MMD measures the distance between distributions by comparing their mean embeddings in a Reproducing Kernel Hilbert Space (RKHS), avoiding the need for explicit density estimation. This mapping to the high-dimensional feature space is implicitly performed by a **kernel function**, and in this work, we employ the classic Gaussian kernel. The MK-MMD loss between all combinations of two distinct tasks, i.e., all task pairs (T_i, T_j), is formulated as:

$$\mathcal{L}_{\text{MK-MMD}} = \sum_{1\leq i<j\leq M} \sum_{k\in\mathcal{K}} \left\| \mathbb{E}_{\mathbf{x}\sim p_{T_i}}\left[\phi(\mathbf{x})\right] - \mathbb{E}_{\mathbf{y}\sim p_{T_j}}\left[\phi(\mathbf{y})\right] \right\|^2_{\mathcal{H}_k}. \tag{9}$$

where $\phi(\cdot)$ is the feature map to the RKHS $\mathcal{H}$ induced by the kernel. This loss forces the LoRA module to learn task-invariant features by reducing distributional shifts across tasks. To enable the computation of cross-task distribution discrepancies within each optimization step, we employ a stratified batch sampling strategy. Specifically, each training batch is constructed to maximize task coverage, aiming to include sub-batches from as many of the M tasks as feasible. This approach ensures that sufficient samples are available to empirically estimate the distribution statistics μ_i and σ_i for the sampled tasks.

Empirically, KL consistently outperforms MMD (Section 4.2). We attribute this to the structured moment-matching bias of parametric KL in the low-dimensional latent space—KL provides per-dimension mean and variance gradient signals ($2r$ for rank r), whereas MMD's kernel-based algorithm aggregates into a single scalar. The closed-form KL also avoids MMD's kernel bandwidth sensitivity. We retain MMD as a complementary, distribution-free validation of the alignment principle.

E.1 Maximum Mean Discrepancy (MMD)

Maximum Mean Discrepancy (MMD)(Sejdinovic et al., 2013) is a kernel-based statistical measure for quantifying the difference between two probability distributions. Given a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ with a characteristic kernel k, the squared MMD between representation p and q is defined as:

$$\text{MMD}^2(p,q) = \|\mu_k(p) - \mu_k(q)\|^2_{\mathcal{H}_k}, \tag{10}$$

where $\mu_k(p) = \mathbb{E}_{\mathbf{x}\sim p}[\phi(\mathbf{x})]$ and $\mu_k(q) = \mathbb{E}_{\mathbf{y}\sim q}[\phi(\mathbf{y})]$ are the mean embeddings of p and q in $\mathcal{H}_k$, and $\phi(\cdot)$ denotes the feature mapping induced by kernel k.

A key advantage of MMD is its ability to capture distributional differences in feature spaces without requiring explicit density estimation. However, its performance heavily depends on the choice of kernel. To address this limitation, the Multiple Kernel MMD (MK-MMD)(Gretton et al., 2012) extends MMD by combining multiple kernels adaptively:

$$\text{MK-MMD}^2(p,q) = \sum_{k\in\mathcal{K}} \|\mu_k(p) - \mu_k(q)\|^2_{\mathcal{H}_k}, \tag{11}$$

where $\mathcal{K}$ is a predefined set of kernels. This variant enhances robustness by combining multiple kernels, allowing the metric to capture multi-scale distributional discrepancies in the reproducing kernel Hilbert space (RKHS).

Table 7: Comparison between VIP-MTL and Align-LoRA.

Dimension	VIP-MTL	Align-LoRA
Target models	Small PLMs (BERT, $\sim$110M)	LLMs (3B–14B)
Core focus	Task imbalance fairness	Isolation vs. alignment paradigm
Method	Probabilistic decoder (adds inference overhead)	Single alignment loss (no architectural change)
Deployment	Non-mergeable	**Fully mergeable (zero inference latency)**
LLM scalability	Unverified	Verified across Qwen2.5, LLaMA2, LLaMA3
Empirical scope	NLU-only	NLU + NLG + reasoning

In the context of transfer learning and domain adaptation, MMD has been widely used as a criterion for aligning feature distributions between source and target domains(Ben-David et al., 2006; Pan et al., 2010). The core idea is to minimize the MMD distance between activations from different domains, encouraging the model to learn domain-invariant representations that generalize well across tasks.

For example, in unsupervised domain adaptation (UDA), MMD is often applied to match the feature distributions of labeled source data and unlabeled target data, reducing domain shift and improving generalization performance. Specifically, given features from the source domain $\mathcal{D}_s = \{\mathbf{x}_i^s\}_{i=1}^{n_s}$ and the target domain $\mathcal{D}_t = \{\mathbf{x}_j^t\}_{j=1}^{n_t}$, the MMD loss is defined as:

$$\mathcal{L}_{\text{MMD}} = \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} \phi(\mathbf{x}_i^s) - \frac{1}{n_t} \sum_{j=1}^{n_t} \phi(\mathbf{x}_j^t) \right\|^2, \tag{12}$$

where $\phi(\cdot)$ denotes the feature embedding function, and the objective is to minimize the distributional discrepancy between the two domains in the shared feature space.

In the context of neural networks for image classification, this MMD loss can be incorporated into the overall training objective alongside the standard classification loss (Long et al., 2015a). The total loss function is typically formulated as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cls}} + \lambda \cdot \mathcal{L}_{\text{MMD}}, \tag{13}$$

where $\mathcal{L}_{\text{cls}}$ is the cross-entropy loss on the labeled source data, and λ is a hyperparameter that balances the contribution of the MMD regularization.

Building on this principle, we propose to incorporate MMD into LoRA for multi-task learning, with a focus on its multiple kernel extension, MK-MMD. Unlike traditional applications that focus on aligning input or hidden-layer features, we apply MK-MMD to the output of the low-rank down-projection matrix $\mathbf{A}$ in LoRA, encouraging the model to learn shared, task-agnostic representations. This design improves multi-task generalization by reducing distributional discrepancies in the representation space, without introducing additional parameters.

While representation alignment has been explored in general multi-task learning, such as VIP-MTL (Hu et al., 2025), our work differs significantly in scope and motivation. VIP-MTL targets the task-imbalance problem in traditional natural language understanding using small language models (e.g., BERT), and its scalability to LLMs remains unverified. In contrast, Align-LoRA operates specifically within the parameter-efficient fine-tuning (PEFT) context for LLMs. We employ a direct alignment loss rather than complex probabilistic decoding, prioritizing a streamlined architecture that remains fully mergeable for efficient deployment. Figure 7 illustrates the difference.

F Theoretical Generalization Bound for Align-LoRA

F.1 Core Conclusion

Align-LoRA aims to improve Multi-Task Learning (MTL) generalization by explicitly minimizing

the distributional discrepancy between task representations. By aligning the representation distributions across multiple tasks (e.g., via MK-MMD), the alignment objective reduces cross-task distributional discrepancy, which tightens the **generalization bound**. The derived generalization bound takes the form:

$$R_{\text{MTL}}(f) \leq \hat{R}_{\text{MTL}}(f) + \frac{\lambda}{M} \sum_{i<j} \Delta(\mathcal{D}_i, \mathcal{D}_j) + \mathcal{O}\left(\sqrt{\frac{M \log(M/\delta)}{n_{\text{total}}}}\right), \tag{14}$$

where $\hat{R}_{\text{MTL}}$ is the empirical training risk, and the distribution discrepancy term $\sum \Delta(\mathcal{D}_i, \mathcal{D}_j)$ is explicitly minimized by the alignment objective.

F.2 Definitions and Notation

To ensure theoretical rigor, we define the following notations consistent with standard domain adaptation literature (Zhao et al., 2025; Chen and Cardie, 2018; Wu, 2024):

- **MTL Scenario:** Let there be M tasks with distributions $\{\mathcal{D}_i\}_{i=1}^M$. The dataset for task i is $\hat{\mathcal{D}}_i$ with size n_i. The total sample size is $n_{\text{total}} = \sum_{i=1}^M n_i$.
- **Centroid Distribution:** $\bar{\mathcal{D}} = \frac{1}{M}\sum_{i=1}^M \mathcal{D}_i$ represents the mixture or centroid of all task distributions.
- **Risks:**
  - Empirical Risk for task i: $\hat{R}_i(f) = \frac{1}{n_i}\sum_{z \in \hat{\mathcal{D}}_i} \ell(f(z))$.
  - Expected Risk for task i: $R_i(f) = \mathbb{E}_{z \sim \mathcal{D}_i}[\ell(f(z))]$.
  - Average MTL Risk: $R_{\text{MTL}}(f) = \frac{1}{M}\sum_{i=1}^M R_i(f)$.
- **Lipschitz Constant:** Let C_L denote the Lipschitz constant of the loss function $\ell(\cdot, \cdot)$ with respect to the representation vectors. This assumption is *local* in the low-rank latent space ($r \ll d$)—it does not require global smoothness of the full LLM loss landscape. In the fine-tuning regime where only 0.2–0.45% of parameters are updated, the model stays near the pre-trained solution, making local smoothness a reasonable standard assumption (Mohri et al., 2018; Long et al., 2015b).

F.3 Derivation of the Generalization Bound

F.3.1 Step 1: Single-Task Generalization Bound

Based on Rademacher complexity theory (Mohri et al., 2018; Bartlett and Mendelson, 2002), for a single task i, with probability at least $1 - \delta'$, the generalization error is bounded by:

$$R_i(f) \leq \hat{R}_i(f) + C_L \cdot \Delta(\mathcal{D}_i, \bar{\mathcal{D}}) + \mathfrak{R}_{n_i}(\mathcal{F}) + \sqrt{\frac{\log(1/\delta')}{2n_i}} \tag{15}$$

where $\Delta(\mathcal{D}_i, \bar{\mathcal{D}})$ measures the distance (e.g., MMD) between the task distribution and the global centroid, and $\mathfrak{R}_{n_i}(\mathcal{F})$ is the empirical Rademacher complexity of the hypothesis class $\mathcal{F}$. To extending this to M tasks simultaneously, we apply the **Bonferroni correction** (utilizing the union bound) (Mohri et al., 2018; Shalev-Shwartz and Ben-David, 2014). We set $\delta' = \delta/M$ to ensure the bound holds for all M tasks with total probability $1 - \delta$.

F.3.2 Step 2: Aggregating to Multi-Task Risk

Averaging Equation 15 over all M tasks:

$$R_{\text{MTL}}(f) \leq \frac{1}{M}\sum_{i=1}^M \hat{R}_i(f) + \frac{C_L}{M}\sum_{i=1}^M \Delta(\mathcal{D}_i, \bar{\mathcal{D}}) + \text{Complexity}(n, M) \tag{16}$$

F.3.3 Step 3: Decomposition of Distribution Discrepancy

We utilize the geometric property of squared metrics (valid for MK-MMD in RKHS). The variance decomposition identity states that the sum of squared distances to the centroid equals the scaled sum of pairwise distances:

$$\sum_{i=1}^M \|\phi(\mathcal{D}_i) - \phi(\bar{\mathcal{D}})\|_{\mathcal{H}}^2 = \frac{1}{M}\sum_{1 \leq i < j \leq M} \|\phi(\mathcal{D}_i) - \phi(\mathcal{D}_j)\|_{\mathcal{H}}^2 \tag{17}$$

Let $\Delta(\mathcal{D}_i, \mathcal{D}_j)$ denote this pairwise discrepancy. Substituting this into the discrepancy term of Equa-

tion 16:

$$\frac{C_L}{M}\sum_{i=1}^{M}\Delta(\mathcal{D}_i,\bar{\mathcal{D}}) = \frac{C_L}{M}\left(\frac{1}{M}\sum_{i<j}\Delta(\mathcal{D}_i,\mathcal{D}_j)\right) = \frac{C_L}{M^2}\sum_{i<j}\Delta(\mathcal{D}_i,\mathcal{D}_j) \tag{18}$$

This step is crucial as it transforms the theoretical bound into an explicitly optimizable pairwise objective.

F.3.4 Step 4: Tightening the Confidence Term

We address the aggregation of confidence intervals. Assuming balanced sample sizes $n_i = n_{\text{total}}/M$:

$$\text{Conf}_{\text{MTL}} = \frac{1}{M}\sum_{i=1}^{M}\sqrt{\frac{\log(M/\delta)}{2n_i}} = \sqrt{\frac{\log(M/\delta)}{2(n_{\text{total}}/M)}} = \sqrt{\frac{M\log(M/\delta)}{2n_{\text{total}}}} \tag{19}$$

Remark: Note that due to the convexity of $f(x) = 1/\sqrt{x}$, the term scales with $\sqrt{M}$. This reflects the independence of the tasks; strictly combining them into a rate of $1/\sqrt{n_{\text{total}}}$ would require stronger assumptions about shared support which we do not make here to maintain rigor.

F.4 Final Bound

Combining the components above, we obtain the final bound for Align-LoRA:

$$R_{\text{MTL}}(f) \le \underbrace{\frac{1}{M}\sum_{i=1}^{M}\hat{R}_i(f)}_{\text{Empirical Risk}} + \underbrace{\frac{C_L}{M^2}\sum_{i<j}\Delta(\mathcal{D}_i,\mathcal{D}_j)}_{\text{Alignment Term}} + \underbrace{\mathcal{O}\left(\sqrt{\frac{M\log(M/\delta)}{n_{\text{total}}}}\right)}_{\text{Confidence Term}} \tag{20}$$

In our optimization objective (Eq. 6 in main text), we define the hyperparameter λ to correspond to the theoretical coefficient $\frac{C_L}{M^2}$. In our practical implementation, we subsume the theoretical constant coefficient $\frac{C_L}{M^2}$ into a user-defined hyperparameter λ, formulated as $\lambda_{\text{impl}} = \frac{\lambda}{M}$ to maintain scalability with task count.

Comparing Align-LoRA to a standard baseline without alignment (Base-LoRA), Align-LoRA explicitly minimizes $\sum_{i<j}\Delta(\mathcal{D}_i,\mathcal{D}_j)$. Since $\Delta_{\text{Align}} \ll \Delta_{\text{Base}}$, the second term in the bound is significantly reduced.

Summary: The analysis offers qualitative support for the discrepancy-reduction principle, explaining why aligning cross-task representations can improve generalization. Empirically, the alignment loss does not converge to zero but stabilizes at a non-trivial level, serving as a continuous regularizer throughout training. Combined with the experimental results above, this provides a principled—though not tight quantitative—rationale for the effectiveness of the alignment approach.

G More result

G.1 Extended Training Analysis

Table 8: Extended training comparison on Qwen2.5-3B (5-task). M-LoRA maintains its advantage over R-LoRA across all training durations, with the gap widening from $+0.78$pp to $+1.23$pp at 3 epochs.

Method	1 Epoch	2 Epochs	3 Epochs
R-LoRA	74.67	74.89	73.95
M-LoRA	**75.45**	**75.61**	**75.18**
Δ (M − R)	+0.78	+0.72	+1.23

G.2 Head Count Ablation

Table 9: M-LoRA performance under varying head counts on Qwen2.5-3B (5-task).

N	2	3	5	8
Avg	74.14	74.62	**75.45**	75.37

G.3 Causal Attribution: Router Removal vs. Dropout Ensembling

M-LoRA differs from R-LoRA and HydraLoRA in two ways: it removes the router, and it retains multi-head dropout. To disentangle these factors, we conduct a 2×2 factorial analysis (Router $\times$ Dropout) on Qwen2.5-3B (5-task). The matrix maps onto existing architectures: R-LoRA represents (*w/ Router, w/ Dropout*), HydraLoRA represents (*w/ Router, w/o Dropout*), and M-LoRA represents (*w/o Router, w/ Dropout*).

Table 10: Full 2×2 factorial (Router $\times$ Dropout) on Qwen2.5-3B (5-task).

	w/ Router	**w/o Router**
w/ Dropout	74.67	**75.45**
w/o Dropout	74.04	**73.82**

Table 11: Multi-seed results on Qwen2.5-3B (8-task, 3 independent runs).

Method	Mean $\pm$ Std
LoRA	76.44 ± 0.21
R-LoRA	79.01 ± 0.24
A-LoRA-K	$\mathbf{80.13 \pm 0.24}$

Two patterns emerge from Table 10. First, removing the router alone (*compare rows*): HydraLoRA drops from 74.04 to 73.58 (w/o Router, Table 1), confirming that router removal without dropout is insufficient. Second, removing dropout (*compare columns*): M-LoRA drops from 75.45 to 73.82 (-1.63pp), a degradation over $3\times$ larger than the router-removal effect (-0.46pp, comparing R-LoRA to HydraLoRA). This confirms that dropout-based collaborative ensembling—not the absence of routing—is the primary driver of M-LoRA's advantage, supporting our interpretation that heads function as collaborators rather than competing specialists. The full 2×2 design thus resolves the potential confound between router removal and dropout ensembling.

G.4 Multi-Seed Validation

To verify that the reported performance gains are not artifacts of random fluctuation, we repeat the main 8-task adaptation experiment on Qwen2.5-3B (Qwen Team, 2024) with three independent random seeds, all other hyperparameters held identical to the main setting (learning rate 0.0002, lora_alpha $= 32$, $\lambda = 0.1$ for A-LoRA-K, dropout 0.1, bfloat16 mixed precision, cosine annealing; full configuration in Appendix K). The same eight tasks and evaluation protocol from Appendix H.3 are used. Table 11 reports the results. The gap between A-LoRA-K and R-LoRA ($+2.09$pp) is approximately $9\times$ the observed standard deviation (0.24pp), and a paired t-test confirms statistical significance ($p < 0.001$). We additionally note that A-LoRA-K outperforms all baselines in every one of $36+$ independent model–benchmark configurations across the main experiments, making random fluctuation an implausible explanation for the consistent gains.

G.5 Performance on Attention structures

To further verify the effectiveness of **Align-LoRA** specifically on **Attention structures** of Qwen2.5, we conducted a supplementary experiment using the exact multi-task setting from **Table 1**. As shown in the table below, when applied to Attention layers, Align-LoRA still achieves significant performance gains over the baseline. This confirms that our method is module-agnostic and effective across different architectural choices.

G.6 Performance on highly dissimilar tasks

This is a crucial question about the limits of our alignment hypothesis. To address this, we ran a supplementary experiment using the exact experimental setup from Section 3 / Table 3.

As detailed in H.2, this dataset (a subset of FlanV2) is specifically designed for high-task diversity, comprising **10 distinct task clusters**. These tasks are highly dissimilar, including not only NLU (e.g., ANLI) and QA (e.g., ARC) but also **Struct-to-Text** (e.g., E2ENLG) and Translation.

We can confirm that **Align-LoRA still demonstrates a significant performance improvement** over the standard high-rank LoRA baseline in this highly dissimilar, multi-domain setting. This provides strong evidence that our alignment strategy is beneficial even when tasks have little in common, as it enhances the learning of foundational shared knowledge.

G.7 Alignment on Multi-Head Architectures

Align-LoRA represents a necessary "course correction" in the field: we demonstrate that for many scenarios, the complex "task-specific" modeling is effectively redundant. Our method achieves superior performance with **zero inference latency**, offering a distinct perspective from the trend of increasing complexity. We think that a future hybrid approach (Align first, then Decompose) is promising. We conducted a new experiment integrating our alignment into the R-LoRA and M-LoRA architecture. As shown below, adding alignment yields further gains even in multi-head structures.

H Datasets

This paper presents a compelling re-evaluation of LoRA-based multi-task learning paradigms, with

Table 12: Performance comparison of different schemes on Qwen2.5's Attention structures

Schemes	QNLI	PiQA	Winogrande	ARC	GSM8K	Avg	%Para
LoRA	82.1	83.2	70.00	88.03	46.60	74.00	0.08
HydraLoRA	82.93	84.02	71.05	87.69	46.25	74.39	0.28
A-LoRA-K	83.34	85.18	72.33	88.15	46.85	**75.17**	**0.08**

Table 13: Performance comparison on highly dissimilar tasks

Metrics	Base	LoRA	HydraLoRA	R-LoRA	M-LoRA	A-LoRA-K
7B	39.82	48.18	49.12	49.51	49.74	**50.09**
14B	45.33	52.74	53.76	54.08	54.18	**54.47**
% Param	-	0.22	0.25	0.25	0.22	0.22

Table 14: Performance Comparison of Alignment Loss Integrated with Multi-Head Architectures (R-LoRA and M-LoRA).

Model	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8
R-LoRA	89.80	62.51	89.36	83.78	95.12	91.02	92.17	50.15
M-LoRA	91.35	62.51	91.98	84.70	95.93	91.02	91.97	50.20
R-LoRA+Align	90.15	62.45	91.68	84.35	95.12	92.17	92.56	50.31
A-LoRA-K	**92.23**	64.85	**92.89**	85.73	95.93	93.35	92.93	53.66
M-LoRA+Align	92.15	**64.93**	92.15	**85.95**	**96.32**	**93.63**	**93.27**	**53.90**

a series of rigorous designs that strongly support its core claims. The setup spans mainstream model families such as LLaMA and Qwen, with parameter scales ranging from 3B to 14B that is sufficiently large for the field of parameter-efficient fine-tuning, and it covers three distinct categories of task datasets. The ablation studies, including the analysis of head similarity, rank scaling effects, and cross-module validation, further validate the rationality of the proposed hypothesis about task-shared representations.

H.1 Experiment 1

In Section 2, we fine-tune Qwen2.5-3B on five tasks:

1. **Natural Language Inference**: QNLI (Wang, 2018)

2. **Physical Question Answering**: PiQA (Bisk et al., 2020)

3. **Word Relation Reasoning**: Winogrande (Sakaguchi et al., 2021)

4. **Closed-Book Question Answering**: ARC (Clark et al., 2018)

5. **Mathematical Reasoning**: GSM8K (Cobbe et al., 2021)

H.2 Experiment 2

In Section 3, following the setting of (Tian et al., 2024; Huang et al., 2023), we utilize a subset of the `Flanv2` dataset (Wei et al., 2021) for complex mixed multi-task and multi-domain scenarios. This dataset covers both Natural Language Understanding (NLU) and Natural Language Generation (NLG) tasks, which are grouped into 10 distinct task clusters. The specific datasets used in our experiments are sourced from LoRAHub (https://huggingface.co/datasets/lorahub/flanv2), a curated and open-sourced repository that provides organized access to these resources. Then we evaluate it with the Big-Bench Hard (BBH) benchmark (Suzgun et al., 2023), using accuracy as the core metric—higher accuracy directly reflects stronger model performance

We summarize the details of the used datasets as follows:

1. **Natural Language Inference**: This task centers on inferring the logical relationship between two sentences, specifically determining whether the second sentence entails, contradicts, or remains neutral relative to the first. The datasets employed include: (1) ANLI; (2) CB; (3) MNLI; (4) QNLI; (5) SNLI; (6) WNLI; (7) RTE.

2. **Coreference Resolution**: This task involves identifying textual mentions that refer to the same entity, thereby demonstrating proficiency in contextual understanding. We utilize the following datasets: (1) DPR; (2) WSC273.

3. **Struct-to-Text Conversion**: This task assesses the ability to generate natural language descriptions from structured data inputs. The datasets used are: (1) CommonGen; (2) DART; (3) E2ENLG; (4) WebNLG.

4. **Closed-Book Question Answering**: This task evaluates models' capacity to answer general knowledge questions without access to external information. We employ the following datasets: (1) ARC; (2) NQ; (3) TriviaQA.

5. **Sentiment Analysis**: As a foundational NLP task, it aims to determine the sentiment polarity (positive or negative) of a given text. The datasets included are: (1) IMDB; (2) Sentiment140; (3) SST-2; (4) Yelp.

6. **Reading Comprehension with Commonsense**: This task integrates traditional reading comprehension with commonsense reasoning, requiring understanding beyond explicit textual content. We use: (1) CosmosQA; (2) ReCoRD.

7. **Paraphrase Detection**: This task demands that models judge whether two sentences convey equivalent meanings, thus indicating semantic equivalence. The datasets utilized are: (1) MRPC; (2) QQP; (3) Paws Wiki.

8. **Translation**: This task involves converting text between languages while preserving the original meaning and subtleties. We employ the following datasets: (1) En-Fr from WMT'14; (2) En-De, En-Tr, En-Ru, En-Fi, En-Ro from WMT'16; (3) En-Es from Paracrawl.

9. **Commonsense Reasoning**: This task assesses the ability to apply physical, scientific principles, and common sense in reasoning processes. The datasets used include: (1) COPA; (2) HellaSwag; (3) PiQA; (4) StoryCloze.

10. **Reading Comprehension**: This task evaluates the capability to derive answers to questions from a provided text containing relevant information. We utilize: (1) BoolQ; (2) DROP; (3) MultiRC; (4) OBQA; (5) SQuADv1; (6) SQuADv2.

H.3 Experiment 3

First, to measure the ability to generalize, models are fine-tuned on the five-task benchmark introduced in Appendix H.1 (QNLI, PiQA, Winogrande, ARC, and GSM8K). Performance is then evaluated on the challenging Big-Bench Hard (BBH) benchmark, which consists of complex reasoning tasks that were not seen during training. We adopt accuracy as the key metric here, with higher accuracy indicating better generalization of the model to these unseen complex reasoning tasks. This setup tests how well knowledge from the training domains is transferred to a different, more difficult set of problems.

Second, to assess the model's ability to adapt to the training domains, we conduct experiments on a broader eight-task benchmark. In this setup, models are evaluated on the corresponding test sets for each of the eight training tasks, with accuracy as the core metric. This evaluation measures the model's in-domain performance, and higher accuracy directly indicates stronger adaptive capability to the training domains. The eight tasks, categorized by their primary reasoning skill, are as follows:

1. **Reading Comprehension**: BoolQ (Clark et al., 2019)

2. **Science Question Answering**: SiQA (Sap et al., 2019)

3. **Physical Commonsense**: PiQA (Bisk et al., 2020)

4. **Word Relation Reasoning**: Winogrande (Sakaguchi et al., 2021)

5. **Commonsense Reasoning**: Hellaswag (Zellers et al., 2019)

6. **Open-Book Question Answering**: OBQA (Mihaylov et al., 2018)

7. **Closed-Book Question Answering**: ARC (easy & challenge) (Clark et al., 2018)

8. **Mathematical Reasoning**: GSM8K (Cobbe et al., 2021)

Finally, for hyperparameter analysis, the model was trained on a 5-task dataset with the same settings as in the Appendix H.1, and the average performance is reported.

I Feature Visualization

We performed t-SNE analysis (Maaten and Hinton, 2008) on the representations of different tasks and compared LoRA with Align-LoRA, as shown in Figure 5. Through representation alignment, Align-LoRA brings the task representations closer to each other, which helps the model learn general knowledge across tasks.

However, the goal is not to make the representations identical. Different tasks inherently contain distinct knowledge, so their representations must maintain a degree of separation to preserve task-specific information. Forcing the representations to become too close can lead to an "over-alignment" problem. This is consistent with our hyperparameter analysis in Figure 4, which demonstrates that an excessively large λ value degrades performance by enforcing an overly aggressive alignment.

[h]

J Baseline

1. **LoraHub** randomly aggregates 20 LoRAs for new downstream tasks. It employs a black-box optimization technique to determine the weight of each LoRA, eliminating the need for gradient calculations of the large model. This involves parameter-level weighted averaging.

2. **LoRA MoE**. A collection of n parameterized experts, denoted as $E_1, \ldots, E_n$, is orchestrated by a router network R. $E_i = B_i A_i$. Router network features a dense layer with adjustable weights W_R from $\mathbb{R}^{d_m \times n}$. A softmax function then processes an intermediate token representation x, yielding gating scores $s_1, \ldots, s_n$ that determine the weighted contribution of each expert's output:

$$s_i = R(x)_i = \text{softmax}(Top(W_R^T x, K)) \tag{21}$$

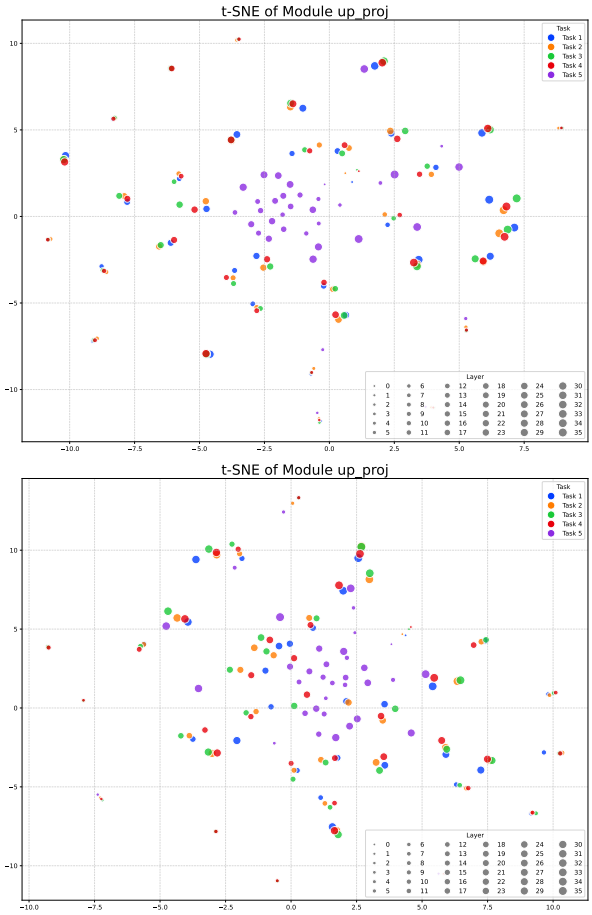

Figure 5: Left: LoRA; Right: Align-LoRA. Through representation alignment, Align-LoRA makes task-specific representations closer to each other, facilitating the model's learning of task-general knowledge.

Subsequently, the overall output y is synthesized by aggregating the Top-K experts' outputs, each modulated by its respective gating score:

$$y = \sum_{i=1}^{n} s_i \cdot E_i(x) \quad \text{(MoE)} \tag{22}$$

This results in a dynamic allocation of the model's capacity, enabling specialized processing by experts as directed by the router's gating mechanism.

3. **HydraLoRA** uses a shared matrix $\mathbf{A}$ and multiple matrices $B_1, \ldots, B_n$. The shared matrix $\mathbf{A}$ is used to project the input vector x into a lower-dimensional space, while each matrix B_i is used to modulate the output of the corresponding expert E_i. The overall output y is synthesized by aggregating the experts' outputs, each modulated by its respective gating

score:

$$y = \sum_{i=1}^{n} s_i \cdot (B_i \cdot A \cdot x) \tag{23}$$

4. **R-LoRA** builds on HydraLoRA by introducing multi-head randomization. It retains a shared projection matrix $\mathbf{A}$ for mapping input vectors x to a lower-dimensional space, while each head matrix B_i undergoes independent random initialization. To promote diversity among head matrices, Dropout is applied to differentiate the input fed into each B_i. The overall output y is generated by aggregating the modulated outputs of all heads through dynamic routing:

$$y = \sum_{i=1}^{n} g_i \cdot (B_i \cdot \text{Dropout}(A \cdot x)) \tag{24}$$

where g_i represents the gating weight for the i-th head, and $\text{Dropout}(\cdot)$ applies stochastic masking to create input variations across different heads.

This approach allows for efficient parameterization and specialization of the model's capacity, leveraging the shared matrix $\mathbf{A}$ for common transformations and the individual matrices B_i for task-specific adjustments.

K Implementation Details

The hyperparameters used for training are as follows: a learning rate of 0.0002, `lora_alpha`=32. For the experiments reported in Table 2, the trainable LoRA components are limited to `q_proj` and `v_proj`—consistent with the setup in HydraLoRA (Tian et al., 2024). Note that for all variants in Table 2 (except the rightmost LoRA†), the LoRA rank is set to 8; the rightmost LoRA† uses a rank of 30, which is adjusted to match the number of trainable parameters across multi-head variants (as indicated by the notation of LoRA† in Table 2). For all other experiments, the trainable LoRA components include `down_proj`, `up_proj`, and `gate_proj`. The number of heads for Table 1 is set to 5, which corresponds to the scenario with 5 tasks. Conversely, the number of heads for Table 2 is set to 10, corresponding to the scenario involving 10 tasks.

Regarding dropout rates: 0.1. The warmup ratio is set to 0.03. For λ in Align-LoRA variants: Align-LoRA-K uses $\lambda = 0.1$, and Align-LoRA-M uses $\lambda = 0.15$.

Mixed-precision training was enabled using bfloat16, with the learning rate scheduler set to cosine annealing. The model was trained on NVIDIA 4090 GPUs.