MTL-LoRA: Low-Rank Adaptation for Multi-Task Learning

Yaming Yang[1*†], Dilxat Muhtar[2*‡], Yelong Shen[3*], Yuefeng Zhan[3], Jianfeng Liu[3], Yujing Wang[3 §], Hao Sun[3], Weiwei Deng[3], Feng Sun[3], Qi Zhang[3], Weizhu Chen[3], Yunhai Tong[1]

[1]Peking University
[2]Nanjing University
[3]Microsoft Corporation
yamingyang@stu.pku.edu.cn, dmuhtar@smail.nju.edu.cn,
{yeshe, yuefzh, jianfengliu, yujwang}@microsoft.com
yhtong@pku.edu.cn

Abstract

Parameter-efficient fine-tuning (PEFT) has been widely employed for domain adaptation, with LoRA being one of the most prominent methods due to its simplicity and effectiveness. However, in multi-task learning (MTL) scenarios, LoRA tends to obscure the distinction between tasks by projecting sparse high-dimensional features from different tasks into the same dense low-dimensional intrinsic space. This leads to task interference and suboptimal performance for LoRA and its variants. To tackle this challenge, we propose MTL-LoRA, which retains the advantages of low-rank adaptation while significantly enhancing MTL capabilities. MTL-LoRA augments LoRA by incorporating additional task-adaptive parameters that differentiate task-specific information and capture shared knowledge across various tasks within low-dimensional spaces. This approach enables pre-trained models to jointly adapt to different target domains with a limited number of trainable parameters. Comprehensive experimental results, including evaluations on public academic benchmarks for natural language understanding, commonsense reasoning, and image-text understanding, as well as real-world industrial text Ads relevance datasets, demonstrate that MTL-LoRA outperforms LoRA and its various variants with comparable or even fewer learnable parameters in MTL setting.

Code — https://github.com/microsoft/MTL-LoRA
Extended version — https://arxiv.org/abs/2410.09437

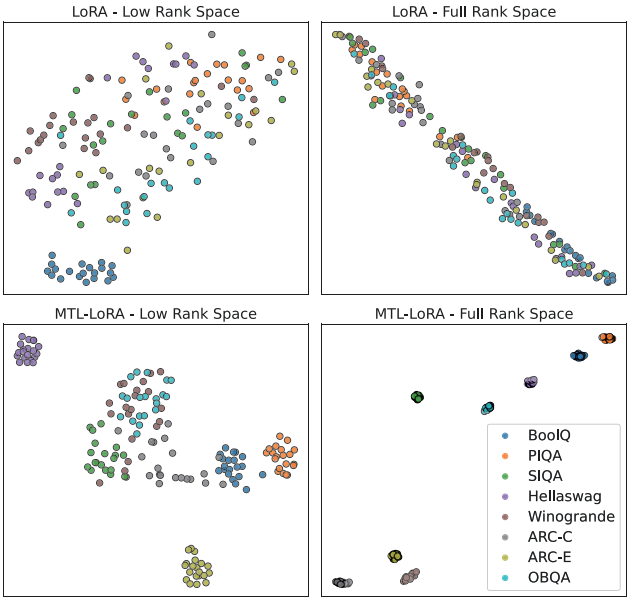

Figure 1: t-SNE visualization of task-specific features extracted from the **O** linear layer of the final block in the LLaMA2-7B model, comparing LoRA and MTL-LoRA after fine-tuning on a commonsense reasoning dataset.

Introduction

Large language models (LLMs) (Brown et al. 2020; Touvron et al. 2023; Team 2023; Yang et al. 2024; Abdin et al. 2024) now drive critical advances in artificial intelligence. By scaling model size and leveraging extensive datasets, LLMs demonstrate exceptional generalization and advanced multi-task capabilities (Wei et al. 2022a; Hoffmann et al. 2022). The concept of "serving one model for different tasks" has led to numerous applications, ranging from natural language processing (Qin et al. 2023) to various domain-specific implementations (Zhao et al. 2023; Wei et al. 2022b; Min et al. 2023). Despite their high generalizability, LLMs still require fine-tuning for specific domains or to update their knowledge base. However, the vast number of parameters in LLMs poses significant challenges regarding computational efficiency and memory consumption during fine-tuning.

Parameter-efficient fine-tuning (PEFT) addresses this challenge by keeping the pre-trained model frozen and fine-

*These authors contributed equally.
†State Key Laboratory of General Artificial Intelligence, School of Intelligence Science and Technology.
‡Work done during internship at Microsoft.
§Corresponding author.

tuning lightweight adapters (He et al. 2021; Houlsby et al. 2019). A prominent PEFT approach is low-rank adaptation (LoRA) (Hu et al. 2021), which trains low-rank "adapter" layers in selected model components. LoRA builds on the insight that fine-tuning updates in pre-trained LLMs exhibit low "intrinsic rank" during task specialization (Aghajanyan, Zettlemoyer, and Gupta 2020), enabling effective approximation through targeted adapters. While LoRA and its recent variants (Liu et al. 2024; Shi et al. 2024; Hayou, Ghosh, and Yu 2024) have shown promise across diverse LLM adaptation scenarios (Liu et al. 2023a; Huang et al. 2023), the growing task complexity and individual fine-tuning costs have spurred research into simultaneous multi-task LoRA training. In this scenario, LoRA projects features from different tasks from a sparse high-dimensional space into a shared, dense low-dimensional space. This projection causes interference and task confusion, amplifying the loss of task-specific information (as shown in the first row of Figure 1). While information sharing is essential in multi-task learning (MTL), it is crucial that different task combinations exchange information distinctly. This necessitates additional design to enable LLMs to adaptively learn varied task information sharing strategies during fine-tuning. Recent LoRA variants have attempted improvements in the multi-task setting, such as ensembling multiple LoRA adapters (Wang et al. 2023) or adopting Mixture of Experts (MoE) structures for soft information specification (Liu et al. 2023b). However, these approaches do not effectively segregate task-specific information or implement distinct information sharing strategies, making them less effective in multi-task scenarios.

In this work, we introduce MTL-LoRA, a LoRA method designed to enhance LLMs with the capability to tackle a variety of tasks in a parameter-efficient manner. MTL-LoRA innovates by implementing task-specific transformations in low-rank space along with a strategy for adaptively exploring multiple information sharing methods. Specifically, MTL-LoRA begins by projecting inputs into lower intrinsic dimensions similar to LoRA. To mitigate the risk of cross-task information interference (Hofmann, Schölkopf, and Smola 2008) within such a condensed space, MTL-LoRA introduces a learnable transformation for each task, ensuring the preservation of task-specific information. Furthermore, acknowledging the role of information sharing among different tasks, especially in enhancing the performance of tasks with limited resources (Crawshaw 2020), MTL-LoRA adopts a dynamic approach to learn different strategies for information sharing. With these improvements, MTL-LoRA efficiently assimilates both task-specific and shared information with minimal trainable parameters. Comprehensive experiments on public academic benchmarks as well as real-world applications demonstrate that MTL-LoRA unleashes the multi-tasking capabilities of LLMs by fine-tuning a limited number of parameters, outperforming LoRA and its variants including.

The key contributions of our work can be summarized as follows:

1 We present MTL-LoRA, which improves the capability of LoRA in MTL through an innovative approach for extracting task-specific information and enhancing cross-task information sharing.

2 Comprehensive experimental evaluations validate the efficacy of MTL-LoRA on both public academic benchmarks and real-world applications.

3 Through extensive ablation experiments and analyses, we confirm the effectiveness of each component of MTL-LoRA and validate the underlying design motivations.

Related Work

Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) adapts large pre-trained models for specific tasks or domains using a small portion of parameters while keeping the main model frozen (He et al. 2021). A popular approach involves inserting trainable, continuous prompts or embeddings into the original text sequence to leverage the base model's knowledge for new tasks (Li and Liang 2021; Liu et al. 2022). Another approach adds additional neural modules, like adapter structures, into pre-trained models (Houlsby et al. 2019; Pfeiffer et al. 2020; Lin, Madotto, and Fung 2020). In this trend, LoRA (Hu et al. 2021) utilizes the concept of low intrinsic dimension (Aghajanyan, Zettlemoyer, and Gupta 2020), introducing two trainable rank decomposition matrices into frozen pre-trained models to estimate the accumulated gradient update during fine-tuning. Due to its lower inference latency and superior performance, LoRA has been widely adopted, and many studies are exploring ways to enhance its efficiency and stability. For example, AdaLoRA (Zhang et al. 2023) incorporates an importance-aware rank allocation method to assign ranks according to layer importance. LoRA+ (Hayou, Ghosh, and Yu 2024) aims to improve LoRA's training stability by using different learning rates for different low-rank matrices. DoRA (Liu et al. 2024) further enhances both the learning capacity and training stability of LoRA by decomposing the pre-trained weights into two components, magnitude and direction, for fine-tuning. While LoRA and its variants show promise in single-task adaptations, their effectiveness diminishes in multi-task scenarios as they update parameters uniformly across all tasks, overlooking crucial task-specific information and dynamic task information sharing. Our work focuses on improving LoRA to acquire both task-specific and task-agnostic knowledge, enhancing its performance in multi-task settings.

Multi-Task Learning

Multi-Task Learning (MTL) aims to optimize all tasks jointly and adapt a single trained model to serve for all tasks (Crawshaw 2020). Since language models trained on large-scale datasets can extract universal representations, previous multi-task learning methods, such as MT-DNN (Liu et al. 2019), typically use a shared pre-trained model with task-specific heads to jointly adapt the model to different tasks. However, as the size of pre-trained models continues to increase, full fine-tuning (FT) introduces

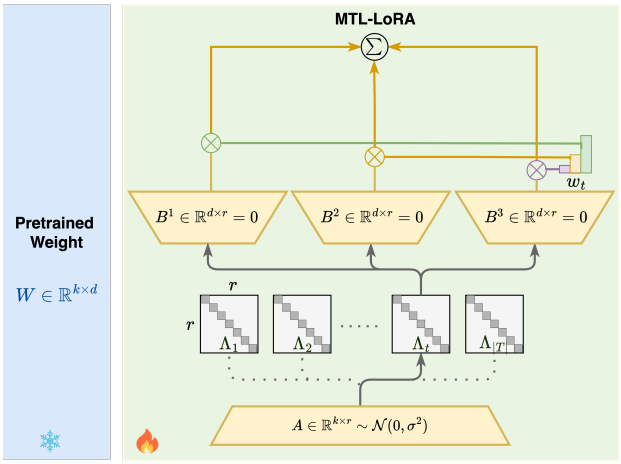

Figure 2: The overall architecture of MTL-LoRA. MTL-LoRA employs task-specific transformation matrices and multiple up-projection matrices to learn both task-specific and shared information.

significant computational overhead and an increased risk of catastrophic forgetting (Biderman et al. 2024). While LoRA offers an alternative to FT, it does not perform as well in multi-task settings (Wang et al. 2023). Approaches like MultiLoRA (Wang et al. 2023) and MoELoRA (Liu et al. 2023b) improve LoRA's multi-task performance in joint training scenarios by integrating multiple LoRAs or utilizing expert routing. However, they fail to strike a good balance between task-specific information and task-information sharing, resulting in suboptimal performance.

Method

In this section, we first introduce the low-rank adaptation method, followed by an in-depth explanation of the proposed MTL-LoRA for multi-task learning.

Low-Rank Adaption

Current LLMs generally follow a decoder-only structure, characterized by a series of blocks, each comprising two key components with residual connections: a multi-head self-attention (MHA) layer and a feed-forward network (FFN) (Brown et al. 2020; Touvron et al. 2023; Team 2023). The MHA layer involves using dense learnable matrices $\mathbf{W}_q$, $\mathbf{W}_k$, $\mathbf{W}_v$, and $\mathbf{W}_o$ to mix the sequence x according to inter-relationships between tokens:

$$\mathrm{MHA}(\mathrm{x}) = \mathrm{Softmax}\left(\frac{(\mathbf{W}_q\mathrm{x})^T\mathbf{W}_k\mathrm{x}}{\sqrt{k}}\right)(\mathbf{W}_v\mathrm{x})^T\mathbf{W}_o, \quad (1)$$

where we assume a single attention head and k denotes the hidden dimension for the head. The FFN layer is usually an MLP with two dense linear projection layers, $\mathbf{W}_{down}$ and $\mathbf{W}_{up}$, and a non-linear activation function $\sigma(\cdot)$ for channel mixing:

$$\mathrm{FFN}(\mathrm{x}) = \sigma(\mathrm{x}\mathbf{W}_{down})\mathbf{W}_{up}. \quad (2)$$

Although LLMs pre-trained with extensive general domain datasets have demonstrated remarkable generalization abilities, there is a need to adapt these models for specific tasks or domains with limited resources. To achieve this, low-rank adaptation (LoRA), inspired by the concept of low intrinsic dimensionality in LLMs, decomposes the weight gradient $\Delta\mathbf{W}$ into low-rank matrices, thereby reducing the number of trainable parameters. Specifically, for a dense weight matrix $\mathbf{W} \in \mathbb{R}^{d\times k}$, LoRA employs two low-rank matrices, $\mathbf{B} \in \mathbb{R}^{d\times r}$ and $\mathbf{A} \in \mathbb{R}^{r\times k}$, to approximate the accumulated gradient updates $\Delta\mathbf{W}$. The rank r is chosen to be much smaller than the minimum of d and k, effectively decreasing the number of trainable parameters. Consequently, the resulting weight matrix is expressed as $\mathbf{W}+\mathbf{BA}$, and the output h for an input x through this updated weight matrix is formulated as:

$$h = (\mathbf{W} + \Delta\mathbf{W})x = \mathbf{W}x + \mathbf{BA}x \quad (3)$$

In implementation, the low-rank matrix $\mathbf{A}$ is initialized with Kaiming Uniform (He et al. 2015) and $\mathbf{B}$ is initialized with zero to ensure $\Delta\mathbf{W} = 0$ at the start, thereby contributing to training stability. A constant scale factor α is also introduced to adjust the magnitude of the changes of the updated matrix $\Delta\mathbf{W}$ made by LoRA modules.

MTL-LoRA

While LoRA effectively fine-tunes LLMs for specific domains using minimal trainable parameters, it does not fully accommodate the dynamics of task-specific and shared knowledge within different tasks or domains, thereby limiting its effectiveness in MTL settings. To address this issue, we introduce MTL-LoRA to improve LoRA with enhanced MTL abilities. The architecture of the proposed MTL-LoRA is detailed in Figure 2. For a given input x_t corresponding to task t, MTL-LoRA projects x_t to low dimension space through $\mathbf{A}$ as in LoRA. However, to enhance the differentiation of tasks within this low, information-dense feature space and maintain task-specific information, MTL-LoRA incorporates a low-rank learnable matrix $\Lambda_t \in \mathbb{R}^{r\times r}$ for each task. This process involves transforming the projected sample $\mathbf{A}\mathrm{x}_t$ via Λ_t to isolate information pertinent to the specific task. Furthermore, we argue that diverse information-sharing strategies are critical for leveraging knowledge from different tasks to improve overall performance. Therefore, rather than relying on a single up-project matrix for information aggregation, we utilize multiple low-rank matrices to explore various information sharing strategies. These combinations are then integrated using a weighted averaging strategy, thereby facilitating adaptive information sharing among different tasks. Specifically, assuming that the up-projection matrix is denoted as $\mathbf{B}^i \in \mathbb{R}^{d\times r}$ and the learnable averaging weight for task t is represented by $w_t \in \mathbb{R}^{n\times 1}$, where n is the number of up-projection low-rank matrices, the output of MTL-LoRA for task t is formulated as:

$$\begin{aligned} h_t &= (\mathbf{W} + \Delta\mathbf{W}_t)x_t \\ &= \mathbf{W}x_t + \sum_{i=1}^{n} \frac{\exp(w_t^i/\tau)\mathbf{B}^i}{\sum_{j=1}^{n}\exp(w_t^j/\tau)}\Lambda_t\mathbf{A}x_t \end{aligned} \quad (4)$$

where τ is a hyperparameter to control the sharpness of the weight distribution, and the superscript represents the in-

Model & Method	# Trainable (%)	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
		Mcc.	Acc	Acc	Acc	Acc	Acc	Acc	Pea.	
MPT-FT	100×8	0.6712	0.9105	0.9044	0.9551	0.9213	0.9036	0.9701	0.9148	0.8939
MPT-LoRA-ST	0.12×8	0.6161	0.9097	0.8578	0.9588	0.9033	0.8881	0.9636	0.8858	0.8729
MPT-LoRA-MT	0.24	0.6528	0.9102	0.8775	**0.9590**	0.9094	**0.9206**	**0.9679**	0.9173	0.8893
MPT-MultiLoRA	0.38	0.6441	0.9092	0.8701	0.9564	0.9085	0.9061	**0.9679**	0.9148	0.8846
MPT-MoELoRA	0.24	0.6421	0.9096	0.8695	0.9570	0.9119	0.9159	0.9639	0.8872	0.8820
MPT-MTL-LoRA	0.24	**0.6754**	**0.9104**	**0.8995**	0.9584	**0.9122**	0.9170	0.9644	**0.9230**	**0.8950**
LLaMA2-FT	100×8	0.7009	0.9144	0.8480	0.9690	0.9273	0.8773	0.9701	0.8965	0.8879
LLaMA2-LoRA-ST	0.12×8	0.6266	0.9099	0.8484	0.9518	0.9049	0.8989	0.9667	0.8915	0.8748
LLaMA2-LoRA-MT	0.24	0.6591	0.9139	0.8603	0.9600	0.9088	0.9170	0.9713	0.9188	0.8887
LLaMA2-MultiLoRA	0.38	0.6134	0.9100	0.8628	0.9552	0.9003	0.9097	0.9633	0.9184	0.8791
LLaMA2-MoELoRA	0.24	0.6366	0.9118	0.8554	0.9568	0.9062	0.9206	0.9656	0.9218	0.8844
LLaMA2-MTL-LoRA	0.24	**0.6797**	**0.9143**	**0.9020**	**0.9628**	**0.9142**	**0.9242**	**0.9713**	**0.9279**	**0.8996**

Table 1: Performance of different adaption methods on the GLUE benchmark, with results reported for the validation set. FT: full parameters fine-tuning. ST: single task fine-tuning. MT: multi-task fine-tuning.

Method	# Trainable(%)	BoolQ	PIQA	SIQA	Winogrande	OBQA	Hellaswag	ARC-E	ARC-C	Avg.
LoRA†	0.83	69.8	79.9	79.5	82.6	81.0	83.6	79.8	64.7	77.6
DoRA†	0.43	**72.0**	83.1	79.9	83.0	81.2	89.1	84.5	71.0	80.5
MultiLoRA	0.40	66.5	65.8	62.8	79.3	75.4	79.2	76.7	59.6	70.7
MoELoRA	0.25	68.0	83.5	70.4	82.5	**83.2**	90.6	86.8	61.5	78.3
MTL-LoRA	0.25	71.0	**84.4**	**80.8**	**84.9**	82.6	**93.1**	**87.0**	**73.4**	**82.1**

Table 2: Commonsense reasoning results. We follow the setting from (Liu et al. 2024; Hu et al. 2023) for jointly training all tasks.† means the results from original DoRA paper (Liu et al. 2024).

dices of the corresponding up-projection matrix and averaging weight. We initialize Λ_i as a diagonal matrix with each diagonal element being 1, thereby ensuring that $\Delta\mathbf{W} = 0$ at the start of training. Building upon these advancements, MTL-LoRA maintains the benefits of parameter efficiency while substantially boosting the MTL capabilities of LoRA.

Experiments

We conduct a series of experiments to demonstrate the effectiveness of MTL-LoRA on various tasks, including natural language understanding (NLU), commonsense reasoning, and image-text understanding. Additionally, we perform ablation studies to illustrate the effectiveness of each component of MTL-LoRA. Finally, we conduct a sensitivity analysis to examine its stability across different hyperparameter configurations.

Evaluation on Public Benchmark

Natural Language Understanding We compare MTL-LoRA against several baseline methods, including full-parameter tuning, single-task fine-tuning with LoRA, multi-task fine-tuning with LoRA, MultiLoRA, and MoELoRA, on both MPT-7B (Team 2023) and LLaMA2-7B (Touvron et al. 2023) models. We use the widely recognized GLUE (Wang et al. 2018) benchmark for evaluation. The GLUE benchmark comprises nine NLU tasks, covering a diverse range of linguistic challenges such as sentiment analysis, textual entailment, and sentence similarity.

To ensure that the LLM with decoder-only architecture generates stable classification results, we adopt the approach of MT-DNN (Liu et al. 2019) by assigning each task its respective classification head. To harness the generative capabilities of LLMs, we reformat each task using a specific template and initialize the weights of the classification head with the corresponding word embeddings of the target answer from the original language model. For each PEFT method, we train only the adapter parameters and the task-specific classification head. Details on hyperparameters and templates can be found in Section A.1 and Section C of the supplementary material.

The results presented in Table 1 demonstrate that MTL-LoRA achieves superior performance, surpassing other baseline methods across both LLMs. Notably, MTL-LoRA not only outperforms the strong MultiLoRA baseline with only 64% trainable parameters but also exceeds the performance of FT while requiring significantly fewer trainable parameters (merely 0.03% per task compared to FT). Furthermore, as shown in Table 1, MTL with LoRA (i.e., LoRA-MT) outperforms single task fine-tuning with LoRA (i.e., LoRA-ST) across all eight tasks and largely reduces the number of adapters that need to be maintained.

While both MoELoRA and MultiLoRA have boosted LoRA's multi-tasking performance, optimizing these models for multi-task scenarios remains challenging, leading to suboptimal performance. In contrast, MTL-LoRA effectively exploits both task-specific and task-agnostic informa-

Model & Method	# Trainable (%)	Task Index														Avg.
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	
		AUC-ROC														
MPT-LoRA-ST	0.12×14	0.8846	0.8361	0.8979	0.8868	0.8806	0.8833	0.8941	0.8589	0.8677	0.8676	0.8490	0.8699	0.8519	0.8689	0.8712
MPT-LoRA-MT	0.24	0.8812	0.8286	0.8863	0.8722	0.8765	0.8827	0.8858	0.8576	0.8584	0.8590	0.8399	0.8680	0.8504	0.8639	0.8650
MPT-MultiLoRA	0.38	0.8874	0.8382	0.8947	0.8871	0.8835	0.8890	0.8944	0.8674	0.8680	0.8712	0.8538	0.8786	0.8598	0.8703	0.8745
MPT-MoELoRA	0.24	0.8860	0.8382	0.8898	0.8837	0.8829	0.8889	0.8950	0.8641	0.8682	0.8720	0.8540	0.8802	0.8576	**0.8769**	0.8741
MPT-7B-MTL-LoRA	0.24	**0.8876**	**0.8384**	**0.9005***	**0.8918***	**0.8840***	**0.8897***	**0.8956***	**0.8678**	**0.8692***	**0.8728***	**0.8561***	**0.8810***	**0.8607***	0.8714	**0.8762**
LLaMA2-LoRA-ST	0.12×14	0.8838	0.8349	0.8992	0.8886	0.8792	0.884	0.8948	0.8575	0.8665	0.8685	0.8495	0.8706	0.8523	0.8703	0.8714
LLaMA2-LoRA-MT	0.24	0.8842	0.8329	0.8932	0.8867	0.8806	0.8859	0.8935	0.8646	0.8652	0.8671	0.8545	0.8763	0.8571	0.8689	0.8722
LLaMA2-MultiLoRA	0.38	0.8879	0.8369	0.8958	0.8922	0.8853	**0.8901**	0.8950	0.8687	0.8693	0.8743	0.8540	0.8808	0.8599	**0.8726**	0.8759
LLaMA2-MoELoRA	0.24	0.8850	0.8370	0.8962	0.8921	0.8850	0.8876	0.8961	0.8685	0.8686	0.8737	0.8550	0.8800	0.8594	0.8722	0.8755
LLaMA2-MTL-LoRA	0.24	**0.8883**	**0.8374**	**0.9016***	**0.8929***	**0.8856**	0.8899	**0.8968***	**0.8688**	**0.8706***	**0.8754***	**0.8582***	**0.8810**	**0.8607***	0.8723	**0.8771**

Table 3: Results of different methods on the test set of Ads dataset. * denotes that the significance test is passed at a 90% confidence level. ST: single task fine-tuning. MT: multi-task fine-tuning.

Method	# Trainable (%)	VQA^{v2}	GQA	NVLR^2	CoCo Cap	Avg.
FT†	100	66.9	56.7	73.7	112.0	77.3
LoRA†	5.93	65.2	53.6	71.9	115.3	76.5
DoRA†	5.96	65.8	54.7	**73.1**	**115.9**	77.4
MTL-LoRA	5.19	**68.6**	**54.9**	72.6	114.6	**77.7**

Table 4: The multi-task evaluation results on VQA, GQA, NVLR^2, and CoCo Caption using the VL-BART backbone. † indicates results taken from the original DoRA paper (Liu et al. 2024).

tion, thereby outperforming LoRA-MT in nearly all tasks.

Commonsense Reasoning In these experiments, we perform a comparison of MTL-LoRA against LoRA and various LoRA variants on LLaMA2-7B for commonsense reasoning tasks. We train each model on eight sub-tasks jointly and evaluate performance on the individual test dataset for each task. Following the same train-test split protocol and instruction prompts as in (Hu et al. 2023; Liu et al. 2024), we report the test set accuracy for each method. Detailed hyperparameter settings can be found in Section A.2 of the supplementary material. Where possible, we report model results as presented in the original papers.

The results in Table 2 show that both MoELoRA and DoRA outperform LoRA and MultiLoRA. We hypothesize that this is because commonsense reasoning tasks require fine-grained task routing or gradient decomposition, rather than merely ensembling different LoRAs. Despite this, MTL-LoRA consistently outperforms all baseline methods. Notably, with only one-third of the LoRA parameters, MTL-LoRA surpasses LoRA by approximately 4%. Moreover, with only half the parameters, MTL-LoRA outperforms the strong baseline DoRA by a large margin of 2%. These results further highlight MTL-LoRA's efficiency and effectiveness in optimizing model performance while minimizing training overhead in multi-task adaptation.

Image-Text Understanding To evaluate the performance of MTL-LoRA in a multimodal multitask fine-tuning context, we compare it with LoRA, DoRA, and FT using VL-BART in four distinct image text tasks. The results for FT, LoRA, and DoRA are taken from the original DoRA paper. For MTL-LoRA, we follow the same settings as DoRA, applying the adapter to the $\mathbf{Q}$ and $\mathbf{V}$ linear layers of the language model. We also unfreeze the bias and layer normalization parameters, training for 20 epochs with the AdamW optimizer and a learning rate of 1×10^{-3}, in line with DoRA's configuration. The rank, alpha, number of up-projection matrices, and temperature for MTL-LoRA are set to 64, 16, 2, and 0.8, respectively.

The results are presented in Table 4. Both MTL-LoRA and DoRA outperform LoRA by 1% with the same or even fewer parameters. Furthermore, MTL-LoRA and DoRA surpass FT while unfreezing only 5-6% of the model's parameters. In the multimodal multitasking scenario, MTL-LoRA also outperforms DoRA with approximately 1% fewer learnable parameters, demonstrating its learning effectiveness in this setting.

Evaluation on In-house Dataset

To further validate the performance of MTL-LoRA in large-scale, complex multi-task scenarios, we compared different multi-task low-rank adaptation strategies on an in-house text Ads relevance dataset (referred to as the Ads dataset).

The text Ads relevance task involves determining whether a query is semantically relevant to a given Ad. This dataset encompasses 14 tasks, covering various production scenarios. The query and Ad pairs are collected from a commercial sponsored search engine, with relevance labels provided by experts. For evaluation, we treat the task as a binary classification problem and report AUC-ROC metrics, following production practices. The dataset consists of 13 million examples in the training set and 2 million examples in the test set, with data collected in multiple languages from global markets. This benchmark is particularly challenging because, while all tasks are from the ad domain, they span different product scenarios. Effectively modeling the correlation between tasks is crucial for achieving optimal performance.

We follow the same experimental design used for fine-tuning in the GLUE benchmark. For further details on the Ads dataset and experimental settings, please refer to Section B and Section A.1 of the supplementary material.

The results are presented in Table 3. In the large-scale Ads dataset, LoRA-MT consistently outperforms LoRA-ST, further underscoring the effectiveness of multi-task fine-tuning. In this context, MTL-LoRA either surpasses or matches

Method	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST2	STSB	Avg.
	Mcc.	Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	Pea.	
LoRA-MT	0.6591	0.9193	0.8603	0.9600	0.9088	0.9170	0.9713	0.9188	0.8887
MTL-LoRA	**0.6797**	0.9143	**0.9020**	**0.9628**	**0.9142**	**0.9242**	**0.9713**	**0.9279**	**0.8996**
w/o Λ_t	0.6567	**0.9164**	0.8971	0.9627	0.9140	**0.9242**	0.9679	0.9261	0.8956 (-0.0039)
w/o τ	0.6592	0.9144	0.8676	0.9606	0.9114	0.9097	**0.9713**	0.9214	0.8895 (-0.0101)
$n = 1$	0.6360	0.9150	0.8725	0.9625	0.9105	0.9170	0.9702	0.9248	0.8886 (-0.0110)

Table 5: Results of ablation studies on MTL-LoRA across from the GLUE benchmark. MT: multi-task fine-tuning.

Method & Model	Task Index														Avg.
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	
	F1														
LLaMA2-LoRA-MT	0.60	0.17	0.58	0.37	0.38	0.52	0.59	0.18	0.61	0.71	**0.44**	0.13	0.18	0.12	0.39
LLaMA2-MTL-LoRA	**0.86**	**0.84**	**0.81**	**0.99**	**0.99**	**0.97**	**0.94**	**0.66**	**0.99**	**0.93**	0.43	**0.73**	**0.34**	**0.55**	**0.79**
MPT-LoRA-MT	0.59	0.27	0.60	0.34	0.36	0.49	0.52	0.30	0.51	0.68	0.26	0.23	0.14	0.14	0.39
MPT-MTL-LoRA	**0.83**	**0.92**	**0.81**	**0.98**	**0.97**	**0.86**	**0.65**	**0.75**	**0.87**	**0.83**	**0.39**	**0.75**	**0.72**	**0.45**	**0.77**

Table 6: Task classification results on Ads dataset using SVM classifier with features extracted by various methods. All results are averaged over five run with different train-test split seed. MT: multi-task fine-tuning.

other methods on all 14 tasks. Notably, while other multi-task adaptations exhibit a seesaw effect when compared to the LoRA-ST approach—improving performance on some tasks but underperforming on others—MTL-LoRA consistently outperforms LoRA-ST across all tasks. This indicates that MTL-LoRA effectively mitigates interference between tasks during adaptation. Furthermore, compared to the second-ranking method in each task, MTL-LoRA achieved statistical significance with confidence 90% in 10 of 14 tasks on the MPT model and 7 of 14 tasks on the LLaMA2 model. This significant performance advantage of MTL-LoRA in the Ads dataset validates its enhanced capability for MTL.

Ablation Study

We conduct ablation studies on MTL-LoRA to assess the impact of three key components: (1) the task-specific learnable transformation matrix Λ_t, (2) the temperature coefficient τ in Eq. 4, and (3) multiple low-rank up-projection matrices n.

For each ablation study, we use LLaMA2-7B as the backbone model and train it on GLUE dataset. In each ablation experiment, we systematically remove or disable one of the key components while keeping all other settings unchanged and report the GLUE metrics. The performance of each modified setting is compared against the full MTL-LoRA configuration. Additionally, we add LoRA's results on multi-task fine-tuning as a reference. Detailed hyperparameter settings are provided in Section A.3 of the supplementary material.

The results, presented in Table 5, clearly illustrate the importance of each component of MTL-LoRA. Omitting any one of these components results in a decline in performance. Notably, the inclusion of multiple low-rank up-projection matrices has the most significant impact. When $n = 1$, the performance of MTL-LoRA falls below that of the LoRA-based multi-task fine-tuning on the GLUE benchmark. This suggests that effectively extracting task-specific information using Λ_t relies on methods that aggregate diverse combinations of information across tasks.

Sensitivity Analysis

In this section, we analyze the robustness of MTL-LoRA under various parameter settings. Our investigation focus on how different values of n, τ, and r impact the performance of MTL-LoRA. We use LLaMA2-7B as the backbone model and conduct comparison on the GLUE benchmark. The results, presented in Figure 3, demonstrate that multiple up-projection metrics are crucial for MTL-LoRA, with $n = 3$ yielding superior results in most scenarios. The value of temperature coefficient also affects the model performance, supporting that different up-projection matrices capture distinct aggregated information. Additionally, the analysis of different values for the rank r reveals that MTL-LoRA maintains robustness at higher rank compared to vanilla LoRA.

Task Differentiation

The primary goal of MTL-LoRA is to enhance the effectiveness of LoRA in multi-task scenarios by preventing cross-task information interference while facilitating task information sharing. Consequently, we assert that the representation outputs of MTL-LoRA should be task-relevant. To validate this, we use the outputs from MTL-LoRA as features for SVM classification, with labels corresponding to their respective tasks. We use the Ads dataset for comparison because it encompasses a larger number of tasks, all within the same Ad domain, thus providing a challenging context for differentiating tasks. Specifically, we randomly sample 1,000 examples for each task from the Ads dataset and use the outputs of different LoRA adapters from the final block of the underlying LLMs as input features. An SVM classifier is trained on 40% of the samples for each method, with the remaining 60% reserved for evaluation. For detailed information on the experimental settings and hyperparameters of

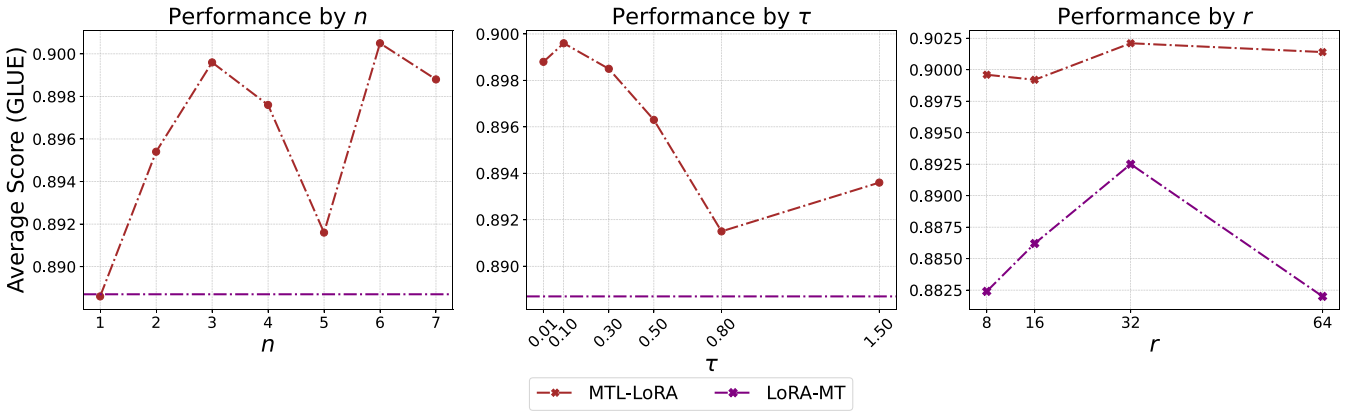

Figure 3: The performance of MTL-LoRA on GLUE benchmark with different hyperparameter configurations.

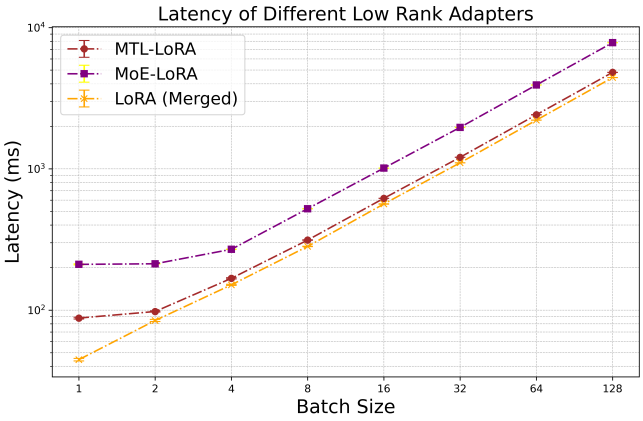

Figure 4: Inference latency of different low-rank adapters on LLaMA2-7B with varying batch sizes. The results are averaged over 100 runs on a single A100-80G GPU. DoRA and MultiLoRA exhibit similar performance to LoRA as they can also be merged into pre-trained weights.

the SVM, please refer to Section A.4 of the supplementary material.

The results, as shown in Table 6, indicate that MTL-LoRA significantly outperforms LoRA in multi-task fine-tuning, achieving a substantial improvement margin with both MPT and LLaMA2. The t-SNE visualization in Figure 1 further supports this observation, revealing that, in the multi-task adaptation scenario, LoRA tends to blend features from different tasks in the low-rank space, whereas MTL-LoRA effectively separates these tasks. When projected back into the full-rank space, MTL-LoRA forms distinct 'task groups', indicating that information is shared within each group while remaining distinct between groups. These findings confirm MTL-LoRA's enhanced ability to distinguish between tasks and effectively reduce task interference. Furthermore, the relatively lower performance of LoRA in this setting highlights its limitations in extracting task-specific information.

Inference Overhead

Although MTL-LoRA demonstrates exceptional performance in multi-task learning, it relies on task-specific information to determine the appropriate transformation matrix for routing (Λ_t in Eq.4). This design choice prevents the merging of all parameters into the original weights as LoRA does, which introduces additional inference latency. However, thanks to MTL-LoRA's task-level routing design, all operations are executed using matrix multiplication, avoiding the expert-level looping required by MoELoRA. This allows MTL-LoRA to fully utilize computational resources without incurring significant inference latency.

To validate this, we compared the inference latency of LoRA, MoELoRA, and MTL-LoRA across different batch sizes on the commonsense reasoning task, as depicted in Figure4. The results clearly demonstrate that MTL-LoRA significantly outperforms MoELoRA in terms of inference speed. Compared to merged LoRA, MTL-LoRA introduces only minimal additional latency during inference, with this difference diminishes further under computationally intensive conditions with larger batch sizes. We believe that in latency-sensitive applications, the gap in inference speed between MTL-LoRA and merged LoRA can be further reduced through system-level optimizations, which we plan to explore in future work.

Conclusion

We propose MTL-LoRA, a novel, advanced parameter-efficient fine-tuning method for multi-task low-rank adaptation. MTL-LoRA enhances the multi-task learning capability of LoRA by incorporating task-specific transformations in low-rank space along with a strategy for adaptively exploring multiple information sharing methods. This approach facilitates the learning of both task-specific and shared information. Comprehensive experiments on multiple public academic benchmarks and a large-scale text Ads relevance dataset demonstrate that MTL-LoRA outperforms LoRA and its variants, including MultiLoRA, MoELoRA, and DoRA, validating its effectiveness in multi-task learning. Furthermore, extensive analysis of intermediate low-rank features and visualizations support our design motivations. For future studies , we will focus on optimizing the design of MTL-LoRA to reduce the additional inference time while maintaining its effectiveness.

References

Abdin, M.; Jacobs, S. A.; Awan, A. A.; Aneja, J.; Awadallah, A.; Awadalla, H. H.; Bach, N.; Bahree, A.; Bakhtiari, A.; Behl, H. S.; Benhaim, A.; Bilenko, M.; Bjorck, J.; Bubeck, S.; Cai, M.; Mendes, C. C. T.; Chen, W.; Chaudhary, V.; Chopra, P.; Giorno, A. D.; de Rosa, G.; Dixon, M.; Eldan, R.; Iter, D.; Goswami, A.; Gunasekar, S.; Haider, E.; Hao, J.; Hewett, R. J.; Huynh, J.; Javaheripi, M.; Jin, X.; Kauffmann, P.; Karampatziakis, N.; Kim, D.; Khademi, M.; Kurilenko, L.; Lee, J. R.; Lee, Y. T.; Li, Y.; Liang, C.; Liu, W.; Lin, E.; Lin, Z.; Madan, P.; Mitra, A.; Modi, H.; Nguyen, A.; Norick, B.; Patra, B.; Perez-Becker, D.; Portet, T.; Pryzant, R.; Qin, H.; Radmilac, M.; Rosset, C.; Roy, S.; Saarikivi, O.; Saied, A.; Salim, A.; Santacroce, M.; Shah, S.; Shang, N.; Sharma, H.; Song, X.; Ruwase, O.; Wang, X.; Ward, R.; Wang, G.; Witte, P.; Wyatt, M.; Xu, C.; Xu, J.; Yadav, S.; Yang, F.; Yang, Z.; Yu, D.; Zhang, C.-Y.; Zhang, C.; Zhang, J.; Zhang, L. L.; Zhang, Y.; Zhang, Y.; and Zhou, X. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. *ArXiv*, abs/2404.14219.

Aghajanyan, A.; Zettlemoyer, L.; and Gupta, S. 2020. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*.

Biderman, D.; Ortiz, J. G.; Portes, J.; Paul, M.; Greengard, P.; Jennings, C.; King, D.; Havens, S.; Chiley, V.; Frankle, J.; Blakeney, C.; and Cunningham, J. P. 2024. LoRA Learns Less and Forgets Less. *ArXiv*, abs/2405.09673.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901.

Crawshaw, M. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

Hayou, S.; Ghosh, N.; and Yu, B. 2024. LoRA+: Efficient Low Rank Adaptation of Large Models. *ArXiv*, abs/2402.12354.

He, J.; Zhou, C.; Ma, X.; Berg-Kirkpatrick, T.; and Neubig, G. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, 1026–1034.

Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; de Las Casas, D.; Hendricks, L. A.; Welbl, J.; Clark, A.; Hennigan, T.; Noland, E.; Millican, K.; van den Driessche, G.; Damoc, B.; Guy, A.; Osindero, S.; Simonyan, K.; Elsen, E.; Rae, J. W.; Vinyals, O.; and Sifre, L. 2022. Training Compute-Optimal Large Language Models. *ArXiv*, abs/2203.15556.

Hofmann, T.; Schölkopf, B.; and Smola, A. J. 2008. Kernel methods in machine learning.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, 2790–2799. PMLR.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Hu, Z.; Wang, L.; Lan, Y.; Xu, W.; Lim, E.-P.; Bing, L.; Xu, X.; Poria, S.; and Lee, R. K.-W. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*.

Huang, C.; Liu, Q.; Lin, B. Y.; Pang, T.; Du, C.; and Lin, M. 2023. LoraHub: Efficient Cross-Task Generalization via Dynamic LoRA Composition. *ArXiv*, abs/2307.13269.

Li, X. L.; and Liang, P. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Lin, Z.; Madotto, A.; and Fung, P. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. *arXiv preprint arXiv:2004.03829*.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023a. Visual Instruction Tuning. *ArXiv*, abs/2304.08485.

Liu, Q.; Wu, X.; Zhao, X.; Zhu, Y.; Xu, D.; Tian, F.; and Zheng, Y. 2023b. When MOE Meets LLMs: Parameter Efficient Fine-tuning for Multi-task Medical Applications. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Liu, S.-Y.; Wang, C.-Y.; Yin, H.; Molchanov, P.; Wang, Y.-C. F.; Cheng, K.-T.; and Chen, M.-H. 2024. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*.

Liu, X.; He, P.; Chen, W.; and Gao, J. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. In Korhonen, A.; Traum, D.; and Màrquez, L., eds., *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4487–4496. Florence, Italy: Association for Computational Linguistics.

Liu, X.; Ji, K.; Fu, Y.; Tam, W.; Du, Z.; Yang, Z.; and Tang, J. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 61–68.

Min, B.; Ross, H.; Sulem, E.; Veyseh, A. P. B.; Nguyen, T. H.; Sainz, O.; Agirre, E.; Heintz, I.; and Roth, D. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2): 1–40.

Pfeiffer, J.; Kamath, A.; Rücklé, A.; Cho, K.; and Gurevych, I. 2020. AdapterFusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*.

Qin, C.; Zhang, A.; Zhang, Z.; Chen, J.; Yasunaga, M.; and Yang, D. 2023. Is ChatGPT a General-Purpose Natural Language Processing Task Solver? *ArXiv*, abs/2302.06476.

Shi, S.; Huang, S.; Song, M.; Li, Z.; Zhang, Z.; Huang, H.; Wei, F.; Deng, W.; Sun, F.; and Zhang, Q. 2024. ResLoRA: Identity Residual Mapping in Low-Rank Adaption. *ArXiv*, abs/2402.18039.

Team, M. N. 2023. Introducing MPT-7B: A New Standard for Open-Source, Commercially Usable LLMs. Accessed: 2023-05-05.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. ArXiv preprint 1804.07461.

Wang, Y.; Lin, Y.; Zeng, X.; and Zhang, G. 2023. MultiLoRA: Democratizing LoRA for Better Multi-Task Learning. *arXiv preprint arXiv:2311.11501*.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35: 24824–24837.

Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; Dong, G.; Wei, H.; Lin, H.; Tang, J.; Wang, J.; Yang, J.; Tu, J.; Zhang, J.; Ma, J.; Xu, J.; Zhou, J.; Bai, J.; He, J.; Lin, J.; Dang, K.; Lu, K.; Chen, K.-Y.; Yang, K.; Li, M.; Xue, M.; Ni, N.; Zhang, P.; Wang, P.; Peng, R.; Men, R.; Gao, R.; Lin, R.; Wang, S.; Bai, S.; Tan, S.; Zhu, T.; Li, T.; Liu, T.; Ge, W.; Deng, X.; Zhou, X.; Ren, X.; Zhang, X.; Wei, X.; Ren, X.; Fan, Y.; Yao, Y.; Zhang, Y.; Wan, Y.; Chu, Y.; Cui, Z.; Zhang, Z.; and Fan, Z.-W. 2024. Qwen2 Technical Report.

Zhang, Q.; Chen, M.; Bukharin, A.; Karampatziakis, N.; He, P.; Cheng, Y.; Chen, W.; and Zhao, T. 2023. AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.