RoLoRA: Fine-tuning Rotated Outlier-free LLMs for Effective Weight-Activation Quantization

Xijie Huang[1], **Zechun Liu**[2*], **Shih-yang Liu**[1], **Kwang-Ting Cheng**[1]
[1]Hong Kong University of Science and Technology, [2]Meta Reality Labs
{xhuangbs,sliuau}@connect.ust.hk, zechunliu@meta.com,timcheng@ust.hk

Abstract

Low-Rank Adaptation (LoRA), as a representative Parameter-Efficient Fine-Tuning (PEFT) method, significantly enhances the training efficiency by updating only a small portion of the weights in Large Language Models (LLMs). Recently, *weight-only* quantization techniques have also been applied to LoRA methods to reduce the memory footprint of fine-tuning. However, applying *weight-activation* quantization to the LoRA pipeline is under-explored, and we observe substantial performance degradation primarily due to the presence of activation outliers. In this work, we propose **RoLoRA**, the first LoRA-based scheme for effective *weight-activation* quantization. RoLoRA utilizes rotation for outlier elimination and proposes rotation-aware fine-tuning to preserve the outlier-free characteristics in rotated LLMs. Experimental results show RoLoRA consistently improves low-bit LoRA convergence and post-training quantization robustness in *weight-activation* settings. We evaluate RoLoRA across LLaMA2-7B/13B, LLaMA3-8B models, achieving up to 29.5% absolute accuracy gain of 4-bit *weight-activation* quantized LLaMA2-13B on commonsense reasoning tasks compared to LoRA baseline. We further demonstrate its effectiveness on Large Multimodal Models (LLaVA-1.5-7B). Codes are available at https://github.com/HuangOwen/RoLoRA

1 Introduction

While we have witnessed the success of Large Language Models (LLMs) such as GPT-4 (Achiam et al., 2023) and LLaMA (Touvron et al., 2023) across various tasks in recent years, the massive model size and expanding training cost for LLMs have necessitated the design of model compression and Parameter-Efficient Fine-Tuning (PEFT) methods. Low-rank Adaption (LoRA) (Hu et al., 2021), as the most favored PEFT method, significantly enhances the fine-tuning efficiency of LLMs.

Recently, quantization techniques, which convert high-precision parameters into lower-bit formats such as INT4, have been integrated with LoRA methods (Dettmers et al., 2024; Li et al., 2024; Xu et al., 2024; Qin et al., 2024). Existing quantization-LoRA schemes can save memory costs during fine-tuning, and some schemes (Li et al., 2024; Xu et al., 2024) can also reduce inference costs by producing quantized LLMs directly. However, these methods only perform *weight-only* quantization, while LoRA *weight-activation* quantization is under-explored. Quantizing both weights and activations in low-bit further saves run-time GPU memory and accelerates compute-intensive matrix-multiplication operations. We observe that 4-bit or 6-bit weight-activation quantization with LoRA finetuning still incurs a high accuracy degradation in LLMs, attributing to the outliers in weight and activation distribution, which stretch the quantization range and increase the quantization error.

Existing methods in the post-training quantization research community have endeavored to tackle the outlier challenge by mixed-precision subgrouping (Zhao et al., 2024; Chee et al., 2024) or shifting outliers from activation to weight (Xiao et al., 2023; Shao et al., 2024). More recently, applying rotation (Ashkboos et al., 2024; Liu et al., 2024c) to the weight matrices of LLMs has demonstrated effectiveness in eliminating activation outliers and keeping computational invariance (Ashkboos et al., 2023a). However, all these methods solve the problems from a post-training perspective, ignoring that outliers will emerge and change distribution during pre-training and fine-tuning (Bondarenko et al., 2021). In this work, we take a step further to utilize the rotation for outliers-removal in LoRA fine-tuning setting and investigate the optimal solution for dynamically integrating rotation with LoRA to preserve the outlier-free characteristics and im-

[*]All the work was done within HKUST and Zechun Liu served an advisory role.

prove weight-activation quantization. Motivated by this target, we propose **R**otated **o**utlier-free **Lo**w-**R**ank **A**daptation (**RoLoRA**), which initially apply in-block and between-block rotation to the pre-trained LLMs, and then utilize rotation-aware fine-tuning to produce outlier-free fine-tuned LLMs as shown in Figure 1. We explore the optimal rotation-aware fine-tuning scheme based on approximation error analysis.

Extensive experimental results prove the effectiveness of RoLoRA across diverse LLMs, tasks, and quantization settings. RoLoRA improves the 4-bit quantization for weights and activations (W4A4) performance up to 14.6 points on the MMLU benchmark compared to LoRA. Compared with existing low-bit LoRA methods, RoLoRA outperforms previous SOTA IR-QLoRA (Qin et al., 2024) with up to 6.0 points on the MMLU benchmark. The proposed RoLoRA is highly efficient with negligible fine-tuning overhead compared to LoRA in the same setting. RoLoRA can also improve the quantization robustness of Large Multimodal Models (LMMs) such as LLaVA (Liu et al., 2024a), and we observe the multimodal understanding is largely retained even after W4A4 quantization as shown in Table 1.

In summary, our work contributes as follows:

- We propose RoLoRA, the first method that explores the feasibility of integrating rotation in LoRA with quantization settings.
- RoLoRA enables robust *weight-activation* quantization of fine-tuned LLMs, especially in low-bit settings such as W4A4 and W6A6.
- We verify the effectiveness of RoLoRA on the LLaMA series (2-7B, 2-13B, 3-8B) across quantizers (RTN/GPTQ), bitwidth (W4A4/W6A6), and benchmarks (Zero-shot commonsense, MMLU). We also demonstrate that RoLoRA can be applied to LMMs.

Question: User: Explain the visual content of the image in great detail. (category: "detail")

GPT-4 Response: The image displays three open pizza boxes with pizzas inside. The first pizza box, situated to the left, contains a smaller pizza. The second pizza box, positioned in the middle, features a larger pizza topped with tortilla chips. The third pizza box, located to the right, contains another smaller pizza. Beside the large pizza, there are two side orders of sauce in bowls, accompanying the delicious meal. The arrangement showcases a delicious spread of pizzas, inviting everyone to enjoy their different flavors and sizes.

LoRA $\xrightarrow{\text{GPTQ}}$ **W4A4:** The image shows a pizza.

RoLoRA $\xrightarrow{\text{GPTQ}}$ **W4A4:** The image features a pizza with a variety of toppings, including cheese, pepperonies, and chicken. There are three pizzas in total, with one of them being a pizza with chicken. The pizzas are placed in a cardboard box, which is open and ready for the pizzas. The pizzas are served in a box, making it a convenient and portable meal option.

Table 1: LLaVA-1.5-7B W4A4 quantization evaluation on the example from LLaVA-Bench.

2 Related Work

Quantization Quantization methods are powerful tools for improving training and inference efficiency. The core insight is replacing full-precision weights and activations with lower-precision representation. Most existing LLM quantization techniques fall in the category of post-training quantization (PTQ) (Liu et al., 2023b; Frantar et al., 2023; Lin et al., 2024; Shang et al., 2024; Chee et al., 2024) that directly quantize the model without extensive training. Among these LLM PTQ methods, most of them apply *weight-only* quantization while few methods explore *weight-activation* quantization (Xiao et al., 2023; Shao et al., 2024; Zhao et al., 2024; Ashkboos et al., 2024). Compared to the *weight-only* quantization, quantizing both weights and activations enables low-precision multiply-accumulation (MAC) units. The core challenge is that outliers in activations cause high quantization errors. This work focuses on the *weight-activation* quantization in the LoRA pipeline.

LoRA Considering that full parameter fine-tuning becomes computationally impractical as the scale of LLM continues to grow, Parameter-Efficient Fine-Tuning (PEFT) methods (Li and Liang, 2021; Hu et al., 2023; Zhang et al., 2023) are designed to reduce the cost by training a relatively small subset of parameters. Low-Rank Adaptation (LoRA) (Hu

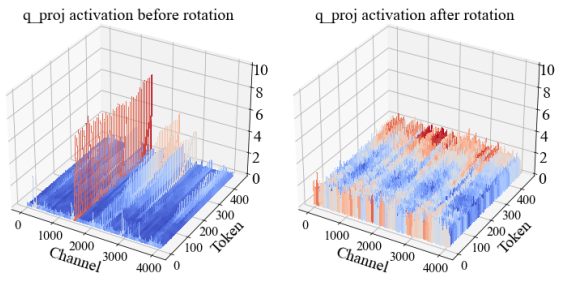


Figure 1: Activation distribution before and after rotation. The visualized input activations are selected from *layers.1.self_attn.q_proj* in LLaMA2-7B.

et al., 2021) is the most adopted PEFT method, considering its flexibility and efficiency. More recently, LoRA variants (Kopiczko et al., 2024; Liu et al., 2024b; Hayou et al., 2024) emerged to improve the effectiveness and efficiency of LoRA. Combining LoRA and quantization (Dettmers et al., 2024) has also been a promising direction as quantization can further save the GPU memory in LoRA finetuning. To further reduce the information distortion of low-bit finetuning, various improvements of QLoRA have been proposed (Xu et al., 2024; Li et al., 2024; Qin et al., 2024). However, these methods only apply quantization to the weight during fine-tuning to reduce memory consumption. This work is the first quantized LoRA scheme that considers the robustness to *weight-activation* quantization.

3 Preliminary and Motivation

3.1 Low-Rank Adaptation (LoRA)

For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, LoRA models the weight update $\Delta W \in \mathbb{R}^{d\times k}$ utilizing a low-rank decomposition, expressed as AB, where $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$ represent two low-rank matrices, with $r \ll min(d, k)$. Consequently, the fine-tuned weight W' can be represented as:

$$W' = W_0 + \Delta W = W_0 + AB, \qquad (1)$$

where W_0 remains static during the fine-tuning process, and the underlined parameters are being trained. Additionally, based on Eq. (1), we can merge the learned ΔW with the pre-trained weight W_0 and obtain W' in advance of deployment, and given that both W' and W_0 both fall within the dimensionality of $\mathbb{R}^{d\times k}$, LoRA and its related variants do not introduce any extra latency during the inference compared to the original model.

3.2 Outlier in Transformer

Starting from small-scale transformer models such as BERT and ViT, researchers have revealed that outliers exist within the weight and activation distribution (Huang et al., 2023; Wei et al., 2022). Their existence in LLMs is also observed in various studies. As shown in the left side of Figure. 1, activation outliers are distributed per channel. While these outliers improve the representative capacity of the transformers (Sun et al., 2024), they bring non-trivial challenges for quantization (Xiao et al., 2023; Liu et al., 2023b).

Most previous solutions to this outlier problem in quantization can be categorized into three types: (1) isolating these outlier values in a sub-group with higher precision, such as LLM.int8 (Dettmers et al., 2022), Atom (Zhao et al., 2024), QuiK (Ashkboos et al., 2023b), and AdaDim (Heo et al., 2024). However, there is non-trivial overhead for the grouping and mixed-precision. (2) shifting the challenge of quantization from activations to weights, such as SmoothQuant (Xiao et al., 2023) and OmniQuant (Shao et al., 2024). However, these methods negatively influence the weight quantization robustness and fail at W4A4 scenarios. (3) rotating activation or weight matrices to remove outliers, such as QuaRot (Ashkboos et al., 2024) and SpinQuant (Liu et al., 2024c). Among these methods, recent rotation-based solutions demonstrate superior effectiveness. However, previous rotation-based methods tackle the outlier challenge from a post-training perspective and have not been explored under PEFT settings.

Thus, it leads to a question: *Can we preserve the outlier-free characteristics of rotated LLMs and benefit from them during PEFT?* We show in this work that we can achieve such a target and step further to investigate the most promising rotation-based fine-tuning solutions in this work.

3.3 Eliminating Outlier with Rotation

A rotation matrix R is defined as an orthogonal matrix with $|R| = 1$, where R also follows the characteristics of the orthogonal matrix that $RR^\top = \mathbf{I}$. If the entries of R are either +1 or -1, it becomes a Hadamard matrix H. Based on the definition, we can efficiently generate H with 2^k entries[1] based on the Hadamard transform (also known as the Walsh–Hadamard transform (Ritter, 1996) as an ex-

[1]For the $n \neq 2^k$ entries, we can also decompose it into $n = 2^k m$ and construct $H_n = H_m \otimes H_{2^k}$ efficiently.

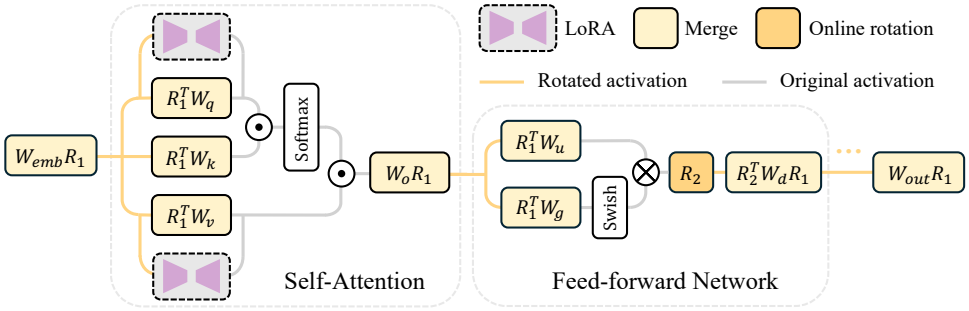


Figure 2: Overview of the proposed Rotated outlier-free LoRA (RoLoRA)

ample of a generalized class of Fourier transforms):

$$H_{2^k} = \begin{bmatrix} H_{2^{k-1}} & H_{2^{k-1}} \\ H_{2^{k-1}} & -H_{2^{k-1}} \end{bmatrix} = H_2 \otimes H_{2^{k-1}}, \tag{2}$$

where $\otimes$ denotes the Kronecker product. The rotation is highly efficient as the matrix-vector product with a $d \times d$ Hadamard matrix H_dX requires $\mathcal{O}(d \log_2(d))$ operations. Previous research (Ashkboos et al., 2023a) has revealed that applying rotation on the weights of *pre-norm* transformers can retain its computational consistency and further lead to fewer outliers in the weight and activation distribution (Ashkboos et al., 2024; Liu et al., 2024c). Concretely, the multiplication of weight matrices with a rotation matrix statistically blends weights with large and small magnitudes together into a more Gaussian-like distribution, thus producing activations with fewer outliers and easier to quantize. The outlier elimination effect of rotation is also theoretically proved in Chee et al. (2024).

4 Method

Motivated by existing challenges of activation outliers and the success of rotation-based solutions (Ashkboos et al., 2024; Liu et al., 2024c), we introduce **R**otated **o**utlier-free **Lo**w-**R**ank **A**daptation (**RoLoRA**). RoLoRA initially apply in-block and between-block rotation to the pre-trained LLMs, and rotation-aware fine-tuning on the rotated LLMs will retain the optimal outlier-free characteristic, producing fine-tuned LLMs highly robust to weight-activation quantization.

4.1 Applying Rotation

Before starting fine-tuning with rotation, we first modify the model to keep computational invariance before and after rotation. First, we need to ensure no scaling operation in the normalization module. For the LLaMA series, this can be implemented by absorbing the RMSNorm scale parameters α into the weight matrix right after the RMSNorm layer (Elhage et al., 2023).

Then, we perform between-block rotation to make sure that the outliers in between-block activation are eliminated. As shown in Figure 2, we classify the weight matrices in LLMs into two groups: *left-side* weights, including W_q, W_k, W_v in self-attention modules, and W_{up}, W_{gate} in feed-forward network modules (which corresponds to the W_u, W_g in Figure 2). *right-side* weights, including W_o in self-attention modules and W_{down} in feed-forward network modules. For the weights of these two groups, we adopt different rotation strategies with

$$W_{\text{left}}^R \leftarrow RW_{\text{left}}, W_{\text{right}}^R \leftarrow W_{\text{right}}R^{-1}, \tag{3}$$

where the rotation R is randomly generated Hadamard matrix. As we also rotated the input X before embedding layer with $X \leftarrow XR^{-1}$ and output Y after *lm_head* with $Y \leftarrow RY$, the final output of the model will be identical to the original model. To avoid overflow issues in the rotation process, we converted the FP16 weights to FP64 and converted them back after the multiplication. The conversion of weight precision is only conducted once at the beginning of the rotation merging and the precision of the rotated weights will keep FP16 during the fine-tuning and inference. There will be no overhead for conversion in the actual inference because the precision during inference is always low-bit (W4A4/W6A6). These rotations are applied before any training and inference, which indicates that there will be no overhead after the merging to original weights.

The rotation that directly applies to weights effectively reduces the outlier in between-block activation, and we refer to the operation as Between-Block Rotation (BBR). Figure. 1 demonstrates the

effect of applying BBR as the activation distribution is smoother and de-centralized. However, another challenge remains that the activation in these modules still suffers from outliers, especially prevalent in FFN as discussed in previous research (Bondarenko et al., 2024). We cannot directly apply rotation similar to BBR because of the non-linear operations such as SwiGLU (Shazeer, 2020) in FFN. To solve this, we adopt the online rotation node before inputting the activation input to W_{down}. This online rotation is implemented following the fast Hadamard kernel (Chee et al., 2024; Ashkboos et al., 2024), which can be seen as a layer dynamically rotating the activation. This online rotation operation is highly efficient as we use the fast Hadamard kernel on CUDA [2], and the overhead is negligible during training and inference. It is referred to as In-Block Rotation (IBR). Note that IBR can also be applied to the self-attention module, but we observe in the experiments of Table 7 that there is no performance improvement with this rotation.

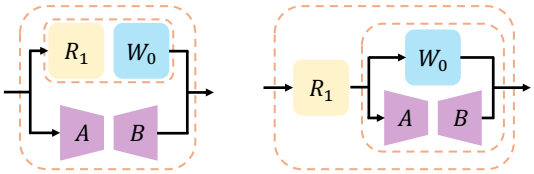
4.2 Rotation-aware Fine-tuning

After performing both BBR and IBR, the between-block and in-block activation outliers are eliminated. This characteristic can lower the quantization error during QLoRA training, enabling a more accurate gradient estimation and smoother optimization for fine-tuning. However, existing research (Bondarenko et al., 2021; Kovaleva et al., 2021) revealed that outliers will change distribution or emerge during fine-tuning and pre-training. This poses a new challenge of dynamically integrating rotation into LoRA to effectively maintain outlier-free characteristics. To design the optimal rotation-aware fine-tuning scheme, we first analyze the approximation difficulty when rotation is applied. We assume that the optimal weight distribution for specific downstream tasks is W^*, and we approximate it with the LoRA weights AB merged with pre-trained weights W_0. The optimization of LoRA fine-tuning could be indicated as

$$\min_{A,B} \|W^* - (W_0 + AB)\|_F, \tag{4}$$

where the $\|\cdot\|_F$ denotes the Frobenious norm. To insert the LoRA module in the rotated models, we propose two rotation-aware fine-tuning schemes, namely LoRA After Rotation (LAR) and LoRA Before Rotation (LBR), as shown in Figure 3.

[2] https://github.com/Dao-AILab/fast-hadamard-transform



(a) LoRA After Roatation (b) LoRA Before Roatation

Figure 3: Two schemes for performing rotation-aware fine-tuning: (a) LAR and (b) LBR.

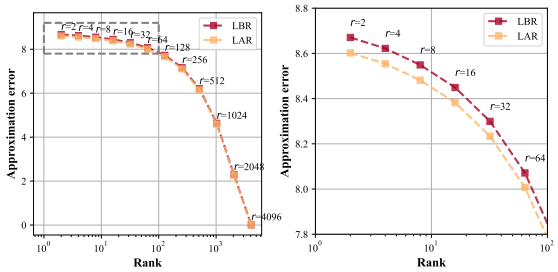


Figure 4: SVD approximation error of optimization targets with different LoRA-rotation integration schemes.

In LAR, we first merge the rotation matrix with pre-trained weights and then use $R_1W_0 + AB$ to approximate W^*. For LBR, we first merge the LoRA weights and rotate them to be $R_1(W_0+AB)$. We assume the optimal weights to be the full-fine-tuning results W_{FT}, and the optimization for these two schemes becomes:

$$\begin{aligned} &\text{LAR: } \min_{A,B} \|AB - O_{\text{LAR}}\|_F, O_{\text{LAR}} = W_{FT} - R_1W_0 \\ &\text{LBR: } \min_{A,B} \|AB - O_{\text{LBR}}\|_F, O_{\text{LBR}} = R_1^{-1}W_{FT} - W_0 \end{aligned} \tag{5}$$

the final optimization is very different. We apply SVD of the approximation target $O_{\text{LAR}}, O_{\text{LBR}} \in \mathbb{R}^{d\times k}$ by $O = USV^T$. The principal singular values and vectors in the first r dimensions are utilized to initialize the LoRA weights with rank r as $A \in \mathbb{R}^{m\times r}$ and $B \in \mathbb{R}^{r\times n}$:

$$A = U_{[:,:r]}\, S^{1/2}_{[:r,:r]} \in \mathbb{R}^{d\times r}, B = S^{1/2}_{[:r,:r]}\, V^T_{[:,:r]} \in \mathbb{R}^{r\times k}. \tag{6}$$

We verify the approximation error of different rank choices r to simulate the LoRA on two rotation schemes. We use a pre-trained LLaMA2-7B as W_0 and a full-parameter fine-tuned model on the Alpaca dataset (Taori et al., 2023) as W_{FT} for the experiments. which is shown in Figure. 4. Based on the results, LAR outperforms LBR in low-rank settings with lower approximation error, suggesting LAR is the better design for rotation-aware fine-tuning. The better approximation indicates that after the two-stage merging with rotation matrices and LoRA weights, the final weights can still retain

the outlier-free property, which is further validated by ablation experiments in Section 5.5.

As a result of the optimal rotation-aware fine-tuning scheme under the LAR setting, we can effectively retain the outlier-free characteristic during LLM fine-tuning, as shown in Figure 5.

5 Experiments

5.1 Settings

Model, LoRA, Quantizer The models for our experiments include LLaMA2-7B/13B (Touvron et al., 2023) and LLaMA3-8B (AI@Meta, 2024). We follow the settings in LLaMA-Factory (Zheng et al., 2024) to implement the training pipeline. The dataset for fine-tuning is Alpaca (Taori et al., 2023) with 52K samples. The weight PTQ methods are the baseline Round-To-Nearest (RTN) and widely used GPTQ (Frantar et al., 2023), and the activation quantizer is RTN across all experiments. We use per-channel symmetric quantization for weights and per-tensor activation quantization.

Tasks Our RoLoRA was verified on seven zero-shot commonsense reasoning tasks using EleutherAI evaluation harness (Gao et al., 2021). These tasks include BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-easy and ARC-challenge (Clark et al., 2018), and OBQA (Mihaylov et al., 2018). Additionally, we also report the accuracy of Massively Multitask Language Understanding (MMLU) benchmark (Hendrycks et al., 2020) for our evaluation.

Baselines We consider two settings for experiments. The first is conducting FP16 fine-tuning with RoLoRA, where we compare the W4A4 and W6A6 quantization results with LoRA. The second is conducting RoLoRA fine-tuning with 4-bit weight quantization, which we refer to as QRoLoRA, and comparing the W4A4 performance with other low-bit LoRA methods including QLoRA (Dettmers et al., 2024), LoftQ (Li et al., 2024), and IR-LoRA (Qin et al., 2024).

5.2 Main Results

We first evaluate RoLoRA against LoRA in FP16 fine-tuning and then apply *weight-activation* PTQ to the fine-tuned LLMs. To ensure a fair comparison, both RoLoRA and LoRA use the same settings (rank, epoch, learning rate, etc.). As listed in Table 2, RoLoRA enhances the quantization robustness of the LLaMA series across various quantization settings on zero-shot commonsense reasoning and MMLU benchmarks. Specifically for the W4A4 low-bit setting, RoLoRA outperforms LoRA with an absolute up to **29.5%** and **14.6%** on ZCSR and MMLU, respectively. Although MMLU contains multiple-choice questions with four options. The relative accuracy below 25% is still meaningful because we observe that some low-bit quantized LLMs cannot even be instructed to give a choice from four options. Our method can better preserve the reasoning performance, thus ensuring most of the time LLaMA is still following the instructions to answer the question rather than generating meaningless tokens. Furthermore, RoLoRA makes it feasible for near-lossless W6A6 quantization of the LLaMa series across multiple tasks.

We further evaluate RoLoRA against QLoRA (Dettmers et al., 2024) and serval baseline methods, including LoftQ (Li et al., 2024), IR-QLoRA (Qin et al., 2024), on 4-bit fine-tuning and then apply W4A4 PTQ to the low-bit fine-tuned LLaMA2-7B. The performance across seven commonsense reasoning tasks and four MMLU subtasks is detailed in Table 3. We can see that RoLoRA consistently improves the performance of the quantized model using the same quantizer. In particular, for W4A4 GPTQ, RoLoRA exceeds QLoRA by **20.5%** on the average accuracy of commonsense reasoning tasks. Across the experiments on both FP16 and 4-bit fine-tuning, we observe that RoLoRA achieves higher performance improvement on the LLMs quantized by GPTQ (Frantar et al., 2023) in general. This observation supports our claim that RoLoRA retains the outlier-free activation in fine-tuning as GPTQ only helps lower the quantization error of weights but not for activation.

5.3 Visual Instruction Tuning

We further verify the effectiveness of RoLoRA on visual instruction tuning tasks with LLaVA-1.5-7B (Liu et al., 2023a), which consists of a language model, Vicuna-7B (Chiang et al., 2023), and a vision encoder CLIP ViT-L-336px (Radford et al., 2021). We finetune the LLaVA-1.5-7B on LLaVA-Instruct-150K[3]. We only perform quantization on the language model and evaluate the LLaVA with quantized Vicuna and full-precision vision encoder on LLaVA-bench (COCO) (Liu et al., 2024a) with GPT-4 (Achiam et al., 2023). The relative score

[3]https://huggingface.co/datasets/liuhaotian/LLaVA-Instruct-150K

Table 2: Comparison of the averaged accuracy on seven Zero-shot Common Sense Reasoning (ZCSR) tasks and MMLU benchmark across LLaMA series. The detailed accuracy for each tasks are listed in Table 10 and Table 11.

#Bits	Quantizer	Method	LLaMA-2 7B		LLaMA-2 13B		LLaMA-3 8B	
			ZCSR[7] Avg.	MMLU[4] Avg.	ZCSR[7] Avg.	MMLU[4] Avg.	ZCSR[7] Avg.	MMLU[4] Avg.
FP16	-	LoRA	68.4	43.5	70.5	52.4	70.0	62.7
W4A4	RTN	LoRA	35.8	23.5	34.4	24.2	36.7	23.3
		RoLoRA	**54.1** (↑18.3)	**25.8** (↑2.3)	**58.7** (↑24.3)	**30.5** (↑6.3)	**50.0** (↑13.3)	**32.1** (↑8.8)
	GPTQ	LoRA	37.0	23.5	34.4	24.4	36.6	23.9
		RoLoRA	**62.3** (↑25.3)	**31.0** (↑7.5)	**63.9** (↑29.5)	**38.9** (↑14.5)	**56.6** (↑20.0)	**38.5** (↑14.6)
W6A6	RTN	LoRA	65.3	35.9	67.3	47.3	67.7	55.3
		RoLoRA	**66.8** (↑1.5)	**40.5** (↑4.6)	**68.4** (↑1.1)	**47.7** (↑0.4)	**67.8** (↑0.1)	**59.4** (↑4.1)
	GPTQ	LoRA	65.5	35.7	68.0	47.6	67.8	54.3
		RoLoRA	**67.1** (↑1.6)	**40.8** (↑5.1)	**68.8** (↑0.8)	**47.9** (↑0.3)	**68.1** (↑0.3)	**59.4** (↑5.1)

Table 3: Comparison of the averaged accuracy of different Low-bit LoRA methods on Zero-shot Common Sense Reasoning tasks and MMLU benchmark on LLaMA2-7B.

#Bits	Quantizer	Method	BoolQ	PIQA	HellaS.	WinoG.	Arc-e	Arc-c	OBQA	Avg.	Hums.	STEM	Social	Other	Avg.
W4A16 ↓ W4A4	RTN	QLoRA (Dettmers et al., 2024)	47.1	51.5	27.5	49.1	28.4	24.6	25.4	36.2	24.1	24.7	22.9	21.8	23.5
		LoftQ (Li et al., 2024)	51.5	50.8	26.6	50.4	27.5	26.0	25.0	36.8	23.9	24.0	22.2	22.2	23.2
		IR-QLoRA (Qin et al., 2024)	45.5	49.7	26.7	50.6	25.7	26.8	26.8	36.0	24.3	24.6	23.9	21.9	23.7
		RoLoRA	59.9	60.5	43.5	51.8	43.7	28.6	28.8	**45.3** (↑8.5)	24.7	25.3	23.6	24.3	**24.5** (↑0.8)
	GPTQ	QLoRA (Dettmers et al., 2024)	51.4	51.6	27.7	51.9	29.6	25.3	26.4	37.7	24.9	24.0	22.2	22.5	23.6
		LoftQ (Li et al., 2024)	55.9	49.2	27.2	49.1	26.6	26.1	24	36.9	24.1	23.8	23.3	22.7	23.6
		IR-QLoRA (Qin et al., 2024)	51.1	49.8	27.6	49.3	27.6	24.6	27.4	36.8	24.6	24.8	22.9	22.7	23.9
		RoLoRA	68.7	73.1	66.8	61.3	61.2	37.8	38.2	**58.2** (↑20.5)	28.3	32.7	32.3	27.2	**29.9** (↑6.0)

across the conversation, detail description, and complex reasoning are reported in Table. 4, where we can observe from the results that RoLoRA help improve the quantization robustness and keep the multi-modal ability during PTQ to the better extent with an increase up to 18.9 overall scores. We also provide an example of the detail description task on a given image shown in Table. 1. While the W4A4 LoRA model only gives a rough superficial description of the images, our W4A4 RoLoRA model fully elaborates the details, such as the toppings and containers.

Table 4: Comparison of the W4A4 quantization performance on LLaVA-Bench of LLaVA-1.5-7B.

#Bits	Quantizer	Method	Conv.	Detail	Reas.	Overall
W4A4	RTN	LoRA	43.2	29.6	31.6	34.9
		RoLoRA	68.8	40.5	51.9	**53.8** (↑18.9)
	GPTQ	LoRA	70.6	41.8	47.9	53.5
		RoLoRA	67.5	48.3	66.2	**60.8** (↑7.3)

5.4 Compatibility with other LoRA variants

We further verify our method on a representative LoRA variant, DoRA (Liu et al., 2024b). DoRA decomposes the pre-trained weight into magnitude and directional components and finetunes both. We also follow this scheme in our rotation-aware fine-tuning stage and refer to this scheme as RoDoRA. As shown in Table 5, RoDoRA achieves 18.3% and 26.7% higher accuracy on W4A4 LLaMA2-7B using RTN and GPTQ as quantizers. The results of RoDoRA also outperform RoLoRA, showing the compatibility of our methods with cutting-edge LoRA variants and potential to further enhance the performance of weight-activation quantization.

Table 5: Compatibility of with DoRA on LLaMA2-7B.

#Bits	Quantizer	Method	ZCSR[7] Avg.
W4A4	RTN	DoRA (Liu et al., 2024b)	36.4
		RoDoRA	**54.7** (↑18.3)
	GPTQ	DoRA (Liu et al., 2024b)	36.6
		RoDoRA	**63.3** (↑26.7)

5.5 Ablation Study and Analysis

When to Apply Rotation? Different from the Rotation-Aware Fine-tuning (RAF) scheme that rotates the LLMs before LoRA fine-tuning, we can also directly apply rotation on an already-finetuned LoRA model. This possible paradigm of LoRA→Rotate→PTQ is referred to as post-training rotation. We evaluate post-training rotation using the same training setting as RoLoRA across the LLaMA series. The W4A4 GPTQ performance on seven zero-shot commonsense reasoning tasks

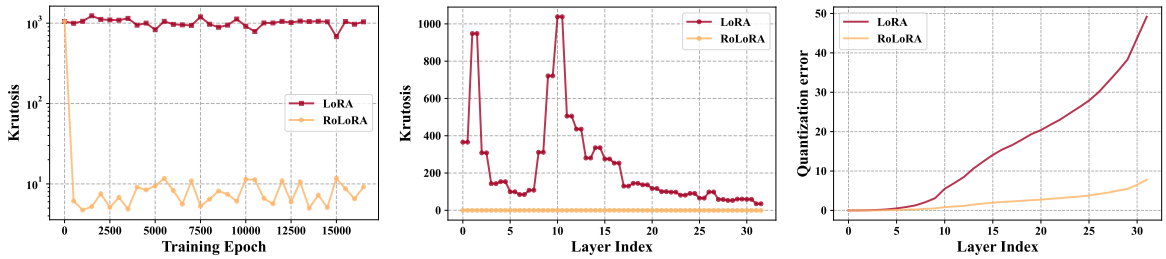


Figure 5: **Left:** The training dynamics of the average Kurtosis of activations, **Middle:** The distribution of Kurtosis of activations across all layers in the final model after fine-tuning with LoRA and RoLoRA, **Right:** The accumulative quantization error of W4A4 GPTQ across all layers in the final model after fine-tuning with LoRA and RoLoRA.

are listed in Table 6. The results indicate that applying rotation before LoRA can consistently enhance the quantization robustness of the fine-tuned LLMs.

Table 6: Ablation on **when** to apply rotation.

Method	LLaMA2-7B	LLaMA2-13B	LLaMA3-8B
RoLoRA	62.3	63.9	56.6
Post-Training Rotation	58.7 (↓3.6)	61.3 (↓2.6)	55.2 (↓1.4)

Where to Apply Rotation? In Figure 2, we introduce two types of rotation in our pipeline, namely Between-Block Rotation applied on all weight matrices and In-Block Rotation applied on *down_proj* in FFN. As discussed in Section 4.1, we can also apply a similar head-wise IBR R_3 for self-attention. The R_3 rotates the W_v and W_o in Figure 2 by $W_v^R \leftarrow W_v R_3, W_o^R \leftarrow R_3^{-1} W_o$. These choices for rotation targets are verified on LLaMA2-7B W4A4 PTQ shown in Table 7. The results suggest that applying and only applying both R_1 and R_2 is the best option to eliminate outliers.

Table 7: Ablation on **where** to apply rotation.

Method	Rotation	ZCSR[7] Avg.
RoLoRA	R_1, R_2	54.1
(−) FFN In-Block Rotation	R_1	40.4 (↓13.7)
(−) Between-Block Rotation	R_2	49.7 (↓4.4)
(+) Attention In-Block Rotation	R_1, R_2, R_3	53.8 (↓0.3)

How to Apply LoRA? In Section 4.2, we propose two rotation-aware fine-tuning schemes LoRA After Rotation (LAR) and LoRA Before Rotation (LBR) shown in Figure 3. We prove that LAR is the better paradigm based on the approximation error analysis compared with full-finetuning. In Table 8, we quantitatively compare the W4A4 quantization performance of two schemes on the fine-tuning of the LLaMA2-7B. The LAR scheme demonstrates better effectiveness, which corresponds to the approximation analysis shown in Figure 4.

Table 8: Ablation on **how** to apply LoRA.

#Bits-Quantizer	Method	ZCSR[7] Avg.	MMLU[4] Avg.
W4A4-GPTQ	LAR	62.3	31.0
	LBR	61.1 (↓1.2)	30.4 (↓0.6)

Outliers Retaining the outlier-free characteristic during LLM fine-tuning is the most important motivation for RoLoRA. To quantitatively validate the effect of outlier elimination, we use kurtosis $\kappa = \frac{\sum_i^k (\mathbf{x}_i - \mu)^4}{\sigma^4 + \epsilon}$ of the activation to measure the outlier presence, where μ and σ are respectively the empirical mean and standard deviation of activation distribution. Generally, a large kurtosis value indicates an activation distribution with heavy tails and a higher likelihood of outliers. We visualize the kurtosis dynamic during fine-tuning with LoRA and RoLoRA in Figure 5. In the early training epochs, the rotation effectively suppresses the activation outliers. The rotation-aware fine-tuning can retain this optimal property. After fine-tuning with RoLoRA, as shown in Figure 5, the kurtosis κ across all layers is significantly reduced, which further gives rise to the low quantization error compared to the LoRA baseline. We also compare the activation distribution of RoLoRA against LoRA across layers in Figure 7 in the Appendix.

LoRA rank settings We explore the robustness of LoRA and RoLoRA towards various rank settings $r \in \{4, 8, 16, 32, 64\}$ when fine-tuning LLaMA2-7B and evaluated on zero-shot commonsense reasoning tasks. The optimal rank setting for RoLoRA and LoRA are 16 and 32, respectively. The lower optimal rank indicates the potential of our RoLoRA to save trainable parameters. Overall, RoLoRA consistently outperforms LoRA regardless of the rank setting, demonstrating its robustness.

Efficiency For the fine-tuning efficiency of

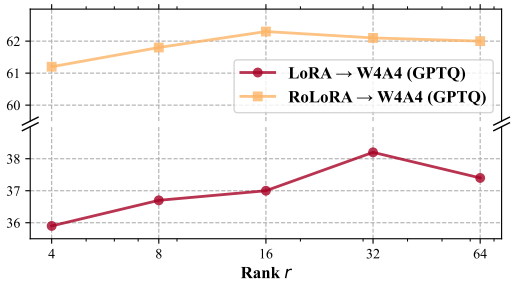


Figure 6: Average accuracy of W4A4 LLaMA2-7B fine-tuned with RoLoRA for varying ranks r.

RoLoRA, the additional training time is only incurred by the online rotation operation (R_2 in Figure 2) as the other rotation (R_1 in Figure 2) can be directly merged into the original weights. There is only one additional matrix multiplication, and the increased rotation parameter can theoretically be considered negligible. We reported the fine-tuning cost of RoLoRA compared to LoRA in the same settings (rank $r = 16$, batch size as 8, 3 total epochs) in Table 9, where RoLoRA significantly improve W4A4 quantized LLaMA2-7B performance with extremely low additional overhead.

Table 9: The fine-tuning costs comparison on LLaMA2-7B with batch size as 8 on NVIDIA H800 80G GPUs.

Method	Training Time	GPU Memory	ZCSR[7] Avg.
LoRA	3.55 h	23.0 GB	37.0 (GPTQ)
RoLoRA	3.65 h	23.1 GB	62.3 (GPTQ)

6 Conclusion

This paper presents RoLoRA, the first work to explore the feasibility of *weight-activation* quantization in LoRA. RoLoRA applies rotation for eliminating outliers in activation distribution and performs rotation-aware fine-tuning to preserve the outlier-free characteristics. We theoretically and empirically investigate how to integrate rotation into LoRA. RoLoRA improves the performance of W4A4 and W6A6 LLMs by a great margin across various tasks with the same training cost. Moreover, RoLoRA can also help visual instruction tuning.

Limitation

In this work, we propose a rotation-based fine-tuning method that can effectively improve quantization robustness to low-bit *weight-activation* PTQ via retaining the outlier-free characteristics. The fine-tuning is conducted on NVIDIA H800 GPUs, while the recent NVIDIA Blackwell-architecture GPUs with 4-bit floating point support may further improve the efficiency. We will take the limitations into account and improve in future work.

Acknowledgement

This research is partially supported by HKSAR RGC General Research Fund (GRF) #16208823. This research is partially supported by ACCESS – AI Chip Center for Emerging Smart Systems, sponsored by InnoHK funding, Hong Kong SAR

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

AI@Meta. 2024. Llama 3 model card.

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2023a. Slicegpt: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*.

Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. 2023b. Towards end-to-end 4-bit inference on generative large language models. *arXiv preprint arXiv:2310.09259*.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. 2024. Quarot: Outlier-free 4-bit inference in rotated llms. *arXiv preprint arXiv:2404.00456*.

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

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2021. Understanding and overcoming the challenges of efficient transformer quantization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7947–7969.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2024. Quantizable transformers: Removing outliers by helping attention heads do nothing. *Advances in Neural Information Processing Systems*, 36.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. 2024. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

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

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

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Nelson Elhage, Robert Lasenby, and Christopher Olah. 2023. Privileged bases in the transformer residual stream. *Transformer Circuits Thread*.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. 2021. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, page 8.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Jung Hwan Heo, Jeonghoon Kim, Beomseok Kwon, Byeongwook Kim, Se Jung Kwon, and Dongsoo Lee. 2024. Rethinking channel dimensions to isolate outliers for low-bit weight quantization of large language models. In *The Twelfth International Conference on Learning Representations*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Xijie Huang, Zhiqiang Shen, and Kwang-Ting Cheng. 2023. Variation-aware vision transformer quantization. *arXiv preprint arXiv:2307.00331*.

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

Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. 2021. Bert busters: Outlier dimensions that disrupt transformers. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

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

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2024. Loftq: Lora-fine-tuning-aware quantization for large language models. In *The Twelfth International Conference on Learning Representations*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023a. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024a. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Shih-yang Liu, Zechun Liu, Xijie Huang, Pingcheng Dong, and Kwang-Ting Cheng. 2023b. Llm-fp4: 4-bit floating-point quantized transformers. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

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*.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. 2024c. Spinquant–llm quantization with learned rotations. *arXiv preprint arXiv:2405.16406*.

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. *arXiv preprint arXiv:1809.02789*.

Haotong Qin, Xudong Ma, Xingyu Zheng, Xiaoyang Li, Yang Zhang, Shouda Liu, Jie Luo, Xianglong Liu, and Michele Magno. 2024. Accurate lora-finetuning quantization of llms via information retention. *arXiv preprint arXiv:2402.05445*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Terry Ritter. 1996. Walsh-hadamard transforms: A literature survey. *Research Comments from Cipers by Ritter*, page 10.

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.

Yuzhang Shang, Zhihang Yuan, and Zhen Dong. 2024. Pb-llm: Partially binarized large language models. In *The Twelfth International Conference on Learning Representations*.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2024. Omniquant: Omnidirectionally calibrated quantization for large language models. In *The Twelfth International Conference on Learning Representations*.

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. 2024. Massive activations in large language models. *arXiv preprint arXiv:2402.17762*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. 2022. Outlier suppression: Pushing the limit of low-bit transformer language models. *Advances in Neural Information Processing Systems*, 35:17402–17414.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, XIAOPENG ZHANG, and Qi Tian. 2024. Qa-lora: Quantization-aware low-rank adaptation of large language models. In *The Twelfth International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Renrui Zhang, Jiaming Han, Chris Liu, Aojun Zhou, Pan Lu, Yu Qiao, Hongsheng Li, and Peng Gao. 2023. Llama-adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *The Twelfth International Conference on Learning Representations*.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. 2024. Atom: Low-bit quantization for efficient and accurate llm serving. *Proceedings of Machine Learning and Systems*, 6:196–209.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*.

A Detailed Evaluation Results

Table 10 and Table 11 listed the full evaluation results on zero-shot commonsense reasoning tasks and MMLU benchmarks, respectively. We use the 'acc_norm' in the evaluation report given by EleutherAI evaluation harness (Gao et al., 2021) as the accuracy if there are such metrics. Otherwise, we use 'acc'.

Table 10: Full accuracy comparison on zero-shot commonsense reasoning tasks of LLaMA series.

#Bits	Quantizer	Method	BoolQ	PIQA	HellaS.	WinoG.	Arc-e	Arc-c	OBQA	Avg.
LLaMA2-7B										
FP16	-	LoRA	81.2	79.8	78.6	70.6	73.9	47.7	46.8	68.4
W4A4	RTN	LoRA	46.0	49.5	27.0	49.6	27.8	24.2	26.8	35.8
		RoLoRA	67.1	67.7	59.7	56.9	58.3	35.0	34.2	54.1
	GPTQ	LoRA	52.3	52.5	26.9	50.4	28.6	25.3	22.8	37.0
		RoLoRA	73.5	76.2	71.8	64.1	67.7	42.2	40.4	62.3
W6A6	RTN	LoRA	76.3	78.0	75.3	69.2	71.2	45.7	41.6	65.3
		RoLoRA	77.9	79.1	76.3	68.5	74.8	47.3	43.6	66.8
	GPTQ	LoRA	76.3	78.2	75.4	69.5	72.1	46.1	40.8	65.5
		RoLoRA	77.4	79.1	76.5	70.4	75.2	47.2	44.0	67.1
LLaMA2-13B										
FP16	-	LoRA	83.9	81.2	80.9	74.2	74.4	51.3	47.6	70.5
W4A4	RTN	LoRA	39.8	52.1	26.1	45.7	25.9	25.8	25.4	34.4
		RoLoRA	70.6	73.9	67.2	59.6	66.8	38.7	34.2	58.7
	GPTQ	LoRA	38.0	50.2	26.0	49.0	25.9	26.4	25.4	34.4
		RoLoRA	74.0	77.2	73.9	66.0	73.3	43.9	38.8	63.9
W6A6	RTN	LoRA	80.8	78.1	77.8	70.3	73.0	49.2	42.2	67.3
		RoLoRA	80.3	78.8	78.0	71.1	77.6	49.6	43.2	68.4
	GPTQ	LoRA	81.9	79.2	78.5	69.3	74.3	51.5	41.2	68.0
		RoLoRA	80.6	79.3	78.1	72.5	77.4	49.4	44.0	68.8
LLaMA3-8B										
FP16	-	LoRA	64.6	82.4	81.4	75.1	81.8	56.5	48.0	70.0
W4A4	RTN	LoRA	46.7	52.2	29.7	47.6	29.3	24.7	26.6	36.7
		RoLoRA	58.0	67.3	57.7	56.0	49.0	30.2	31.8	50.0
	GPTQ	LoRA	42.5	54.4	29.4	49.0	31.1	22.5	27.0	36.6
		RoLoRA	63.2	71.1	66.7	60.2	60.3	38.2	36.8	56.6
W6A6	RTN	LoRA	75.5	78.3	77.4	70.8	76.4	51.2	44.0	67.7
		RoLoRA	78.6	79.5	76.7	71.1	77.6	49.8	40.8	67.8
	GPTQ	LoRA	77.9	78.3	77.9	71.3	75.2	50.5	43.2	67.8
		RoLoRA	78.1	79.3	76.8	71.9	76.7	50.9	42.8	68.1

B Hyper-parameters for Reproduction

In Table 12, we list the detailed hyper-parameters for reproducing RoLoRA and LoRA results. We do not apply searches on any hyperparameters for better accuracy, all the settings for the LLaMA series and LLaVA align with the default settings of Zheng et al. (2024) and Liu et al. (2024a).

C Activation Distribution Visualization

We visualize the magnitude of the activation of fine-tuned LLaMA2-7B using LoRA and RoLoRA in Figure 7. The visualizations reveal a noticeable amount of outliers presented in the LoRA fine-tuned model, but are highly eliminated in RoLoRA counterpart.

Table 11: Full accuracy on MMLU Benchmark of LLaMA series.

#Bits	Quantizer	Method	Hums.	Other	Social	STEM	Avg.
LLaMA2-7B							
FP16	-	LoRA	41.5	50.8	48.2	34.7	43.5
W4A4	RTN	LoRA	24.2	24.8	22.7	21.7	23.5
		RoLoRA	24.7	26.2	27.2	25.7	25.8
	GPTQ	LoRA	24.3	24.5	23.0	22.0	23.5
		RoLoRA	30.1	33.0	32.0	29.4	31.0
W6A6	RTN	LoRA	35.4	40.6	37.5	30.4	35.9
		RoLoRA	38.2	45.4	44.7	35.2	40.5
	GPTQ	LoRA	34.2	39.4	39.4	30.6	35.7
		RoLoRA	37.8	46.1	46.2	34.9	40.8
LLaMA2-13B							
FP16	-	LoRA	49.6	59.2	59.9	42.8	52.4
W4A4	RTN	LoRA	25.0	25.7	23.4	22.4	24.2
		RoLoRA	28.9	32.5	33.2	28.4	30.5
	GPTQ	LoRA	25.5	24.2	24.1	23.4	24.4
		RoLoRA	37.7	42.3	43.7	32.7	38.9
W6A6	RTN	LoRA	44.3	52.8	55.0	38.6	47.3
		RoLoRA	45.0	52.9	55.2	39.1	47.7
	GPTQ	LoRA	44.8	54.7	53.8	39.0	47.6
		RoLoRA	45.6	53.7	55.2	38.7	47.9
LLaMA3-8B							
FP16	-	LoRA	57.4	70.7	72.8	52.7	62.7
W4A4	RTN	LoRA	23.6	24.3	23.7	21.8	23.3
		RoLoRA	30.8	34.5	33.5	30.5	32.1
	GPTQ	LoRA	24.6	23.0	23.4	24.3	23.9
		RoLoRA	36.0	42.2	43.6	33.5	38.5
W6A6	RTN	LoRA	49.7	63.0	64.4	47.2	55.3
		RoLoRA	52.7	67.5	70.0	51.1	59.4
	GPTQ	LoRA	48.8	61.8	63.9	45.7	54.3
		RoLoRA	52.9	68.3	69.6	50.4	59.4

Table 12: Detailed hyper-parameters for fine-tuning different LLMs and LMMs.

Model	LLaMA2-7B	LLaMA2-13B	LLaMA3-8B	LLaVA-1.5-7B
Epoch	3	3	3	1
Batch Size (Per GPU)	8	4	8	2
Gradient Accumulation	1	2	1	64
Warmup Ratio	0.01	0.01	0.01	0.03
Optimizer	AdamW	AdamW	AdamW	AdamW
LoRA Rank r	16	16	16	128
LoRA Dropout	0	0	0	0.05
LoRA Target	W_q, W_v	W_q, W_v	W_q, W_v	$W_q, W_k, W_v, W_o, W_u, W_d, W_g$
Learning Rate	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$	$2e^{-4}$

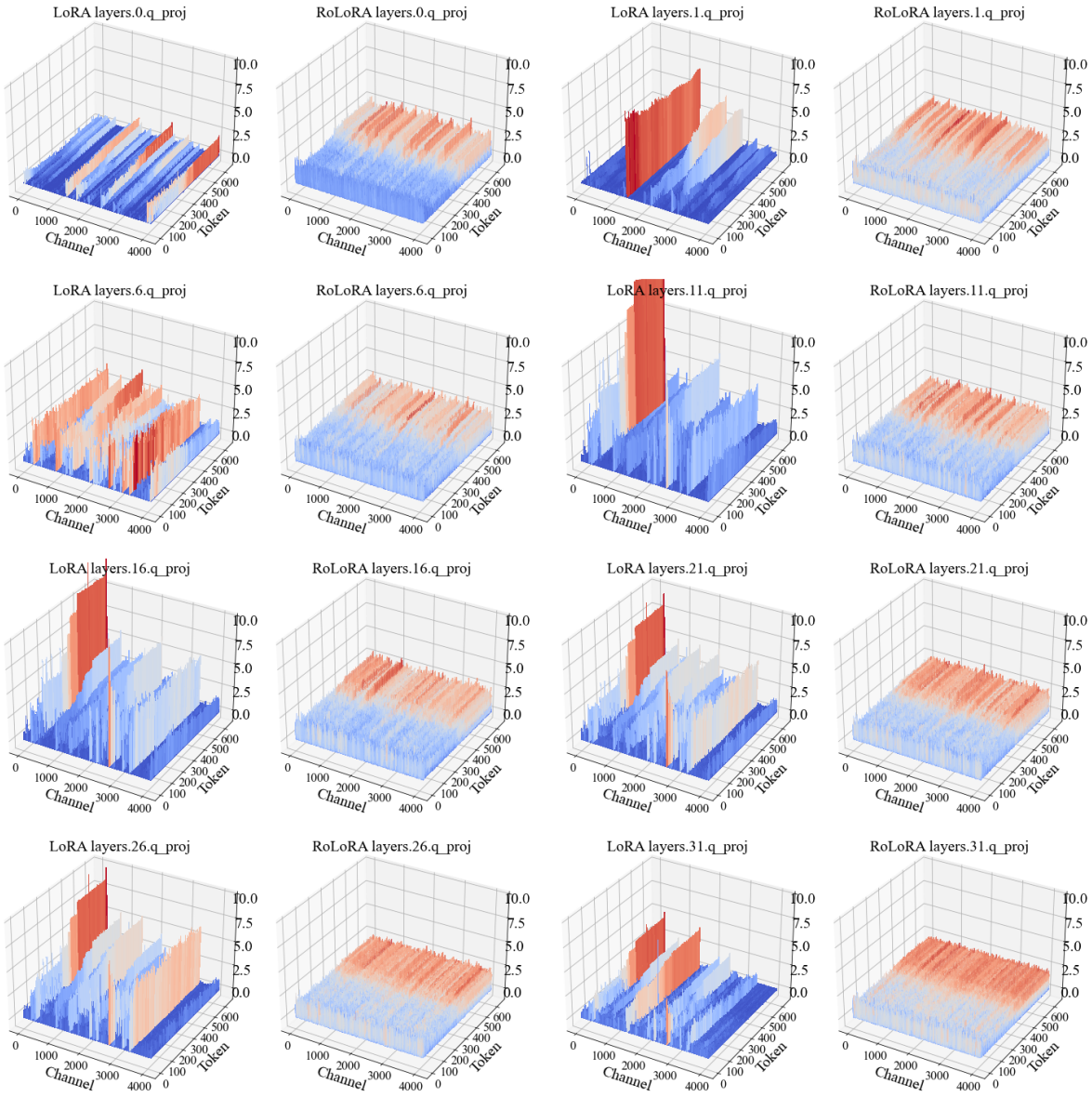


Figure 7: Final activation distribution of the fine-tuned model produced using RoLoRA and LoRA. We select the output activation of *q_proj* across layers with the index of 0, 1, 6, 11, 16, 21, 26, 31.