Low Rank Quantization Adaptation for Large Language Model

Haocheng Sun[1*], **Wang Chen**[1*], **Haoxiang Wu**[1*], **Yuhui Zeng**[1*]

[1]**AI class**, Class 1, Master's Program of 2024, Institute of Artificial Intelligence, Xiamen University, China
{36920241153249, 36920241153198, 36920241153256, 36920241153277}@stu.xmu.edu.cn

Abstract

As the parameters of Large Language Models (LLMs) increase, quantization has emerged as a potent strategy for model compression and acceleration. Concurrently, Low-Rank Adaptation (LoRA) has been recognized as an effective method for enhancing LLM performance. However, integrating LoRA with quantization presents significant challenges, particularly in preserving the quantization format after model optimization. In this paper, we introduce Low rank Quantization Adaptation (LoQA) for LLM, a novel approach that effectively fine-tunes holistic quantization parameters. Specifically, we first propose a new perspective of quantization operator, which is compatiable with LoRA and mathematically equivalent to the original operator. In this way, **all** the parameters (scale and zero point) are finetuned simultaneously, and thus yields notable improvements in model performance. Thanks to the expanded optimization landscape, LoQA is broadly applicabile to various Post-Training Quantization (PTQ) techniques, ensuring better generalizability in practical deployments. To maintain the stability of the optimization, we further propose a LoRA scaling strategy that leverages quantization data to adjust the norm of the low rank adaptation, regulating the speed of convergence in optimization and preventing inappropriate LoRA scaling, which could lead to overfitting or underfitting. Compared to existing methods, LoQA consistently achieves performance gains across a wide range of models, proving its effectiveness and adaptability.

Introduction

In recent years, large language models (LLMs) (Zhang et al. 2022; Le Scao et al. 2023; Brown et al. 2020; Touvron et al. 2023a,b) have demonstrated remarkable performance across various fields, attracting significant attention. However, the increasing number of parameters in these models has made training and fine-tuning progressively more challenging. This has led to a research focus on efficiently enhancing model performance on diverse tasks using massive datasets, thereby facilitating the deployment and utilization of LLMs by researchers and the general public.

Parameter-efficient fine-tuning (Xu et al. 2023; Hu et al. 2021; Köksal et al. 2023; Liu et al. 2024) and quantization (Xiao et al. 2023; Lin et al. 2023; Frantar et al. 2022; Shao et al. 2023; Ma et al. 2024) have emerged as prominent methods for improving training efficiency and compressing models. Parameter-efficient fine-tuning techniques aim to minimize the number of fine-tuning parameters and computational complexity. These techniques enhance model performance while reducing fine-tuning costs, time, and computational resource consumption. For example, QLORA efficiently fine-tunes a 65B parameter model on a 48GB GPU using Low Rank Adapters and innovative 4-bit quantization (Dettmers et al. 2023). The low-rank adaptation (LoRA) (Hu et al. 2021) method reduces the number of fine-tuning parameters through low-rank matrix multiplications. This approach decreases memory usage during gradient updates and accelerates training speed. Additionally, freezing parameters in the backbone network during optimization allows for the integration of quantization methods. Mapping backbone network parameters to low-bit representations further improves training efficiency. A series of post-training quantization methods (Frantar et al. 2022; Xiao et al. 2023; Lin et al. 2023; Frantar et al. 2023; Shao et al. 2023; Ma et al. 2024) can quickly produce high-performance low-bit quantized models for the backbone network. The integration of quantization and parameter-efficient fine-tuning presents substantial challenges within neural network optimization. Notably, maintaining the quantized format of the backbone network proves difficult following the integration of fine-tuned parameters. Initially, QLoRA (Dettmers et al. 2024) addresses this issue by employing post-training quantization to preserve the structure post-fusion. However, this method partially compromises the precision of fine-tuned parameters, impacting the overall accuracy of the model. To tackle this, QA-LoRA (Xu et al. 2023) constrains the dimensions of low-rank matrices, allowing the fine-tuning parameters to be incorporated directly into the zero points of the quantized backbone network. This ensures the stability of the quantization fixed points during parameter fusion, although it restricts the optimization space for fine-tuned parameters, thus capping potential performance gains for the language model.

In response, this paper introduces a novel approach named Low-Rank Quantization Adaptation (LoQA). This method enhances all quantized parameters with an efficient fine-tuning module. Conceptually, if the quantization zero points in the backbone network are viewed as translational op-

[*]These authors contributed equally.

erations on intra-group weight parameters, the scale parameters then serve as scaling transformations that adapt these parameters to the quantization range. LoQA comprehensively optimizes both sets of quantization parameters through gradient-based methods, thereby broadening the optimization space. Concurrently, it preserves the quantized structure of the backbone network, ensuring that the quantization fixed points remain stable. The fine-tuning of the two sets of quantization parameters under a low-rank framework minimizes both time and computational expenses, yielding an optimized quantized model efficiently.

Moreover, the increase in learnable parameters introduces the risk of overfitting, especially in smaller datasets. To mitigate this risk, we also introduce a LoRA scaling technique that regulates the optimization process of the adaptive low-rank matrices and uses quantization data to regularize the fine-tuning parameters. Loss optimization curves demonstrate that LoRA scaling effectively moderates the convergence rate, thus preventing overfitting in smaller datasets. Extensive experiments with various quantization configurations on different benchmarks and datasets using the LLaMA model consistently demonstrate performance improvements, validating the effectiveness of LoQA. In summary, our contributions are as follows:

- **Providing a new perspective in quantization.** While existing methods primarily focus on the fine-tuning of shallow quantization parameters merged learned parameters without preserving the underlying quantized structure, our Low-Rank Quantization Adaptation (LoQA) expands the optimization space for fine-tuning all quantized parameters. By conducting a thorough analysis of the dequantization process, our method efficiently fine-tunes all quantized parameters, effortlessly enhancing the model's capacity.
- **We have devised a noval LoRA scaling strategy.** This strategy is designed to mitigate overfitting phenomena during the fine-tuning phase. This strategy employs a novel approach by using quantization statistics to regularize adaptive low-rank matrices, thus ensuring controlled and predictable model convergence. The effectiveness of the LoRA scaling method is substantiated by optimization loss curves, which illustrate its capacity to maintain training stability and model generalizability.
- Empirically, our methodologies have shown substantial improvements in performance across a wide range of models and quantization configurations. Fine-tuning various models on the extensive Flan-v2 dataset has yielded consistent enhancements, demonstrating the practical efficacy and broad applicability of our proposed techniques. For instance, the performance of the LLaMA7B model, quantized at 3-bit width, improved by 2.8% in the MMLU 5-shot tasks compared to the existing state-of-the-art. This underscores the robust capacity of LoQA. Furthermore, our experiments on smaller datasets trained over multiple epochs have even slightly surpassed the current state-of-the-art, illustrating the effectiveness of our LoRA scaling strategy in preventing overfitting.

Related Work

Parameter-Efficient Fine-Tuning (PEFT)

Parameter-efficient fine-tuning techniques are designed to minimize the number of fine-tuning parameters and computational complexity, making them a potent strategy for fine-tuning Large Language Models (LLMs). For instance, the Low-Rank Adaptation (LoRA) method (Hu et al. 2021) reduces the number of fine-tuning parameters through low-rank matrix multiplications, decreasing memory usage during gradient updates and accelerating training speeds. LoRA achieves comparable performance to full fine-tuning with significantly fewer learnable parameters across various tasks. Additionally, other types of parameter-efficient fine-tuning methods, such as DoRA (Liu et al. 2024), VeRA (Kopiczko, Blankevoort, and Asano 2023), and PiSSA (Meng, Wang, and Zhang 2024), have also been proposed.

Quantization of LLMs

As the parameters of Large Language Models increase, quantization has emerged as a powerful strategy for model compression and acceleration. Existing quantization methods primarily focus on preserving or restoring the accuracy of quantized LLMs during the inference stage (Zhu et al. 2023), aiming to reduce memory usage and computational costs without retraining the LLM. One of the main challenges is dealing with outliers in the parameters and activations, which can cause significant errors during quantization. Particularly in LLMs, outliers significantly complicate the quantization process. GPTQ (Frantar et al. 2022) employs progressive quantization and mitigates quantization errors by applying Hessian matrix corrections to the full-precision parameters. AWQ (Lin et al. 2023) addresses the issue of activation outliers by scaling the input channels of both weights and activations. Furthermore, OmniQuant (Shao et al. 2023) introduces a post-training quantization algorithm that leverages gradient optimization by making the scaling and clipping parameters learnable, thereby improving the adaptability and performance of the quantized model.

Joint Efficient Fine-Tuning and Quantization

The integration of quantization and parameter-efficient fine-tuning poses significant challenges in neural network optimization, especially in maintaining the quantized format after fine-tuning. This work aims to extend the goals of parameter-efficient adaptation and computation-efficient tuning and deployment, which can further enhance the efficiency and scalability of LLMs while mitigating the negative impacts of quantization errors. Previous work, such as QLoRA (Dettmers et al. 2023), failed to maintain the quantized format after fine-tuning. QA-LoRA (Xu et al. 2023) performs group-wise operations on quantization and low-rank adaptation, which preserves the quantized format post-fine-tuning, but the optimization space is limited to translational operations on intra-group weight parameters, showing limited generalizability on larger datasets.

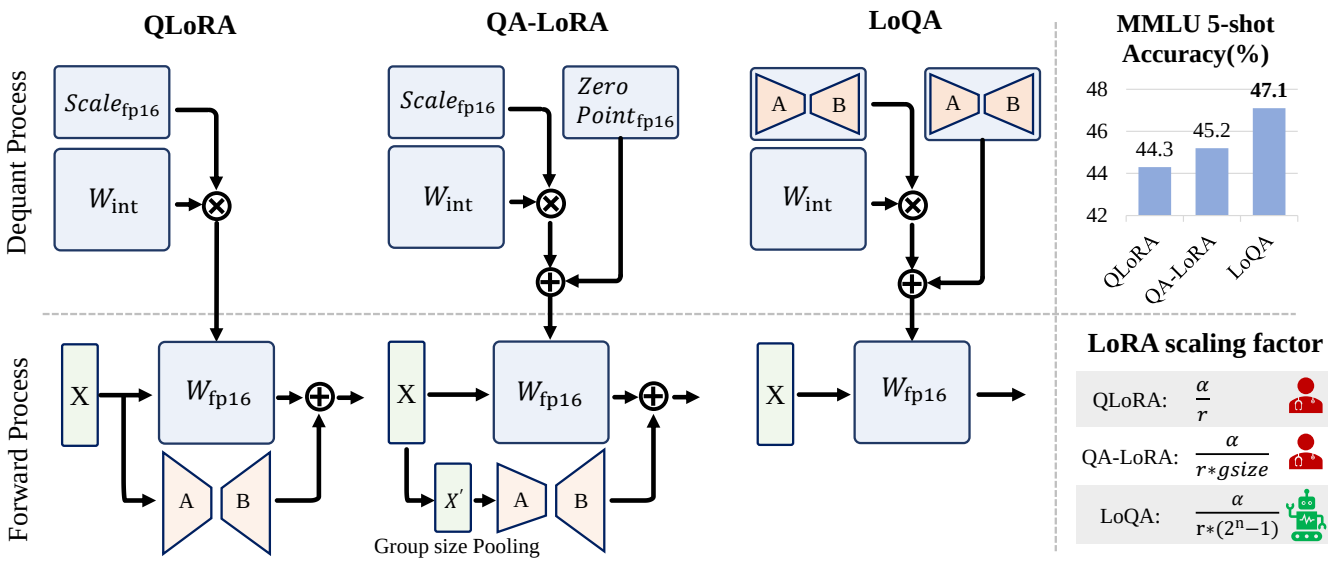

Figure 1: LoQA (Low-rank Quantization Adaptation) is an efficient method that combines fine-tuning with quantization, differing from QLoRA and QA-LoRA methods by integrating low-rank adaptation during the dequantization process. This integration ensures that quantization is completed simultaneously with fine-tuning, preserving the quantized structure. In downstream inference tasks, LoQA maintains the advantageous properties of quantization without requiring additional parameters or increasing inferential overhead.

Low-rank Quantization Adaptation

LoQA, as illustrated in Figure 1, seeks to expand the optimization space and preserve the quantization format by applying low-rank adaptation adjustments to all quantized parameters and integrating them with the adaptive matrix and the original quantized weights. It initially demonstrates the feasibility of efficiently expanding the optimization space, both theoretically and practically, as elaborated in Section . Section explores LoQA's adaptability to various Post-Training Quantization (PTQ) methods. In practical training, it was observed that employing traditional LoRA scaling strategies with LoQA often led to overfitting or underfitting issues; thus, a new LoRA scaling strategy grounded in considerations of numerical stability is introduced in Section .

Preliminaries

To make it clear, we follow the symbolic notation system to elucidate the Low-Rank Adaptation (LoRA) methodology (Hu et al. 2021). Specifically, we consider $\mathbf{W}$ as a matrix representing the pretrained weights for a specific layer, with dimensions $D_{\text{out}} \times D_{\text{in}}$. Features are represented by a vector $\mathbf{x}$ of length D_{in}. Consequently, the output vector $\mathbf{y}$, having dimensions D_{out}, is computed as $\mathbf{y} = \mathbf{W}\mathbf{x}$. Central to the LoRA approach is the introduction of two low-rank matrices, $\mathbf{A}$ and $\mathbf{B}$, which are dimensioned as $D_{\text{int}} \times D_{\text{in}}$ and $D_{\text{out}} \times D_{\text{int}}$ respectively. The term D_{int}, significantly smaller than both D_{in} and D_{out}, ensures that the product $\mathbf{BA}$ is a low-rank matrix yet aligns in size with $\mathbf{W}$.

In the training phase, the computation is augmented to include a scaling coefficient s, yielding the formula $\mathbf{y} = \mathbf{W}\mathbf{x} + s \cdot \mathbf{B}\mathbf{A}\mathbf{x}$. This formulation allows $\mathbf{W}$ to remain static, while $\mathbf{A}$ and $\mathbf{B}$ are updated to enable efficient parameter tuning. Post-training, we employ the reparametrized weight matrix $\mathbf{W}' = \mathbf{W} + s \cdot \mathbf{BA}$, which is utilized during inference to compute the output as $\mathbf{y} = \mathbf{W}'\mathbf{x}$, facilitating accelerated computation.

The paradigm of amalgamating quantization with fine-tuning, such as QLoRA (Dettmers et al. 2023), generally begins with a post-training quantization technique to generate a quantized model. To illustrate this concept straightforwardly, let us consider the application of a standard min-max quantization method. Assuming a model with weights in FP16 format (denoted as $\mathbf{W}^{\text{FP16}}$), and aiming to quantize these to N bits, the quantization is governed by the following formula:

$$
\begin{aligned}
\mathbf{W}^{\text{Int4}} &= \text{clamp}\left(\text{round}\left(\mathbf{Temp}\right), 0, 2^N - 1\right), \\
\mathbf{Temp} &= \frac{\mathbf{W}^{\text{FP16}} - \mathbf{ZeroPoint}^{\text{FP16}}}{\mathbf{Scale}^{\text{FP16}}} \\
\mathbf{ZeroPoint}^{\text{FP16}} &= \mathbf{W}^{\text{FP16}}_{\text{min}}, \\
\mathbf{Scale}^{\text{FP16}} &= \frac{\mathbf{W}^{\text{FP16}}_{\text{max}} - \mathbf{W}^{\text{FP16}}_{\text{min}}}{2^N - 1}.
\end{aligned}
\tag{1}
$$

In the above equation, $\mathbf{Scale}^{\text{FP16}}$ represents the quantization step size, and $\mathbf{ZeroPoint}^{\text{FP16}}$ serves as the offset, or zero points, facilitating the alignment of real and quantized values. The function $\text{clamp}(z, r_1, r_2)$ is used to restrict the value of z within the range defined by r_1 and r_2, effectively bounding it by returning r_1 if z is less than r_1, and r_2 if z

exceeds r_2. The function round(z) adjusts z to the nearest integer.

In the quantization framework, $\mathbf{W}^{\text{FP16}}$ and $\mathbf{W}^{\text{Int4}}$ matrices are both dimensioned as $D_{\text{out}} \times D_{\text{in}}$, whereas $\mathbf{Scale}^{\text{FP16}}$ and $\mathbf{ZeroPoint}^{\text{FP16}}$ are defined for each output channel and shaped as $D_{\text{out}} \times \frac{D_{\text{in}}}{\text{groupsize}}$. The parameter groupsize indicates the division of input dimensions into groups for localized quantization.

This quantization procedure involves storing the values of $\mathbf{W}^{\text{Int4}}$, $\mathbf{ZeroPoint}^{\text{FP16}}$, and $\mathbf{Scale}^{\text{FP16}}$. To revert to the floating-point representation $\mathbf{W}^{\text{FP16}}$ during inference, we deploy the corresponding dequantization process:

$$\tilde{\mathbf{W}}^{\text{FP16}} = \mathbf{W}^{\text{Int4}} \odot \mathbf{Scale}^{\text{FP16}} + \mathbf{ZeroPoint}^{\text{FP16}}, \quad (2)$$

In the above equation, $\tilde{\mathbf{W}}^{\text{FP16}}$ serves as an approximation of the original weight matrix $\mathbf{W}^{\text{FP16}}$. This approximation facilitates the restoration of the floating-point values from their quantized integer form, enabling the use of lightweight models in high-precision tasks.

Preserving Holistic Quantization Format in Adaptation

While the dequantization process may incur a slight loss of precision, it preserves the substantial benefits of quantization, including notably reduced storage requirements and minimized input/output (I/O) overhead. Quantization necessitates only the storage of the compressed values $\mathbf{W}^{\text{Int4}}$ and a reduced number of instances of grouping parameters. Methods like QLoRA ((Dettmers et al. 2022)) and GPTQ-LoRA commonly leverage this technique to minimize memory usage and accelerate the post-quantization fine-tuning phase. Nevertheless, these models often do not fully capitalize on the potential synergies that could arise from a more cohesive integration of quantization and fine-tuning. A key obstacle is the challenge associated with effectively reincorporating the learned low rank adaptations back into the original quantized models. To elucidate why such integration is complex, consider the following equations that describe the forward process using these methods:

$$\mathbf{W}' = \tilde{\mathbf{W}} + s \cdot \mathbf{BA} = (\mathbf{W}^{\text{Int4}} \odot \mathbf{Scale}^{\text{FP16}} + \mathbf{ZeroPoint}^{\text{FP16}}) + s \cdot \mathbf{BA}. \quad (3)$$

In this model, $\mathbf{W}^{\text{Int4}}$ maintains its dimensions as $D_{\text{out}} \times D_{\text{in}}$, which is compatible with the dimensions of $\mathbf{BA}$. However, the integer nature of $\mathbf{W}^{\text{Int4}}$ presents challenges for seamless integration with the floating-point operations involving $\mathbf{BA}$, potentially leading to type conflicts and precision issues.

Within the QA-LoRA framework, although a straightforward integration of learned low rank adaptations into $\mathbf{W}^{\text{Int4}}$ is impractical due to datatype discrepancies and possible precision degradation, an effective strategy has been devised. This method incorporates LoRA modifications within $\mathbf{ZeroPoint}^{\text{FP16}}$, thus avoiding any precision loss. However, this approach encounters a challenge due to dimensional disparities: $\mathbf{ZeroPoint}^{\text{FP16}}$ is structured as $D_{\text{out}} \times \frac{D_{\text{in}}}{\text{groupsize}}$, which does not align with the dimensions of $\mathbf{BA}$, set at $D_{\text{out}} \times D_{\text{in}}$. This discrepancy poses significant integration challenges, necessitating further adaptations or modifications to harmonize these components within the quantization and adaptation framework.

To address the dimensional misalignment in the QA-LoRA framework, two pragmatic modifications are implemented:

- **Reshape $\mathbf{A}$**: The shape of matrix $\mathbf{A}$ is altered to $D_{\text{int}} \times \frac{D_{\text{in}}}{\text{groupsize}}$ to align with the dimensions of $\mathbf{ZeroPoint}^{\text{FP16}}$.
- **Adjust Input $\mathbf{x}$**: The shape of the input vector $\mathbf{x}$ is adjusted to $\frac{D_{\text{in}}}{\text{groupsize}}$, enabling alignment with the reshaped $\mathbf{A}$. This adjustment is achieved using an average pooling layer sized to the groupsize, effectively reducing the dimensionality of $\mathbf{x}$ to match $\mathbf{A}$.

These modifications ensure that the integration of low rank adaptations into the quantized structure is seamless and does not compromise fidelity. The revised merge equation, which incorporates LoRA adjustments within $\mathbf{ZeroPoint}^{\text{FP16}}$, is formulated as follows:

$$\mathbf{ZeroPoint}^{\text{FP16}} = \mathbf{ZeroPoint}^{\text{FP16}} + s \cdot \mathbf{BA}. \quad (4)$$

This equation confirms that the integration of low rank adaptations maintains the original quantization precision of the weights. The adjustments are localized to $\mathbf{ZeroPoint}^{\text{FP16}}$, preserving both the integrity and efficiency of the quantized model. Furthermore, this approach facilitates enhanced performance through fine-tuning adaptations, thereby maintaining the efficacy of the model even in resource-constrained environments. This methodology not only preserves the quantized format but also ensures the continued effectiveness of the model. Upon reviewing Equation 3, an intriguing possibility emerges: if learned LoRA adaptations can be integrated into $\mathbf{ZeroPoint}^{\text{FP16}}$, is it feasible to similarly integrate these adaptations into $\mathbf{Scale}^{\text{FP16}}$? To investigate this, we propose a method analogous to the one used in QA-LoRA, employing the Hadamard product (element-wise multiplication) between $\mathbf{BA}$ and $\mathbf{W}^{\text{Int4}}$. This concept is reflected in the following modification to the equation:

$$\mathbf{W}' = \tilde{\mathbf{W}} + s \cdot \mathbf{BA} = (\mathbf{W}^{\text{Int4}} \odot \mathbf{Scale}^{\text{FP16}} + \mathbf{ZeroPoint}^{\text{FP16}}) + \mathbf{W}^{\text{Int4}} \odot s \cdot \mathbf{BA}. \quad (5)$$

While theoretically sound, this method raises practical efficiency concerns due to the computational overhead of additional operations on $\mathbf{W}^{\text{Int4}}$, typically stored as Int32. To address this, we propose LoQA (Low-Rank Quantization Adaptation), which achieves equivalent adjustments during dequantization by fine-tuning $\mathbf{Scale}^{\text{FP16}}$ and $\mathbf{ZeroPoint}^{\text{FP16}}$. Algorithm pseudocode 1 outlines this efficient adjustment, focusing on modifying the dequantization process. Specifically, this involves adjusting the value of scale before proceeding with the forward process, as shown in Equation 6.

$$\mathbf{Scale}'^{\text{FP16}} = \mathbf{Scale}^{\text{FP16}} + s \cdot \mathbf{BA},$$
$$\mathbf{y} = \tilde{\mathbf{W}}\mathbf{x} = (\mathbf{W}^{\text{Int4}} \odot \mathbf{Scale}'^{\text{FP16}} + \mathbf{ZeroPoint}^{\text{FP16}})\mathbf{x}. \tag{6}$$

LoQA not only facilitates these adjustments during the dequantization phase but also maintains efficiency by optimizing $\mathbf{Scale}^{\text{FP16}}$ and $\mathbf{ZeroPoint}^{\text{FP16}}$.

Algorithm 1: LoQA Pseudocode in the PyTorch-like style

```
# D_in, D_out, D_int: the input, output, and adaptation dimensions
# L: the quantization group numbers ( D_in // L is the group size)
# s: the coefficient for adaptation; N: the bit width of quantization
# zp_lora_A, zp_lora_B, scale_lora_A, scale_lora_B: LoRA Parameters
# loqa_scaling: the coefficient for LoQA-S

def forward(W_quant,scale,zeropoint):
    scale_new = scale + loqa_scaling * scale_lora_B @ scale_lora_A
    zp_new = zp + s * zp_lora_B @ zp_lora_A
    W_tilde = scale_new * W_quant + zp_new #Eq.(7)
    result = W_tilde @ x

def deployment(scale,zeropoint):
    scale = scale + s * scale_lora_B @ scale_lora_A
    zeropoint = zp + s * zp_lora_B @ zp_lora_A
    del zp_lora_A, zp_lora_B, scale_lora_A, scale_lora_B
    return scale,zeropoint
```

Scaling Strategy of LoQA

The scaling strategy in LoRA and its derivative, QLoRA, is typically expressed as $\frac{\alpha}{rank}$, where 'rank' denotes the dimensionality D_{int} of the low-rank matrices, and α serves as a hyperparameter. In QA-LoRA, the scaling is modified to $\frac{\alpha}{rank \times \text{groupsize}}$, a formula driven by the need to adjust the quantized zero point across a group of parameters. This division by the groupsize is designed to enhance numerical stability.

However, within the context of LoQA, our empirical observations suggest that neither the groupsize-adjusted LoRA scaling nor the conventional LoRA scaling is optimal. High scaling values frequently result in overfitting and instability due to excessive numerical magnitudes, whereas low values tend to induce underfitting due to insufficient numerical impact. The challenge, therefore, is to stabilize these scaling values within LoQA-S, as illustrated in Equation 5. LoQA-S involves pointwise multiplication of the $\mathbf{BA}$ matrix with $\mathbf{W}^{Int4}$, where the fixed points in $\mathbf{W}^{Int4}$ can unpredictably amplify the LoRA values, leading to numerical instability.

A logical solution might be to normalize each quantization group by dividing by the largest fixed point in that group. However, this approach would require storing and computing the maximum quantized values for different group fixed points, which would substantially increase both storage and computational demands. In LoQA-S, we propose a more efficient, approximate solution. Our adapted LoRA scaling employs the formula $\frac{\alpha}{rank \times \text{maxq}}$, where maxq is calculated based on the bit width; for instance, in N-bit quantization, maxq $= 2^N - 1$, shown in Figure.1 This strategy not only enhances efficiency but also strikes a commendable balance between maintaining data stability and optimizing model fitting capabilities.

Flexibility and Compatibility of LoQA

Recent advancements in Post-Training Quantization (PTQ) have introduced methods that further compress the ZeroPoint by converting it from floating-point to integer format, as highlighted in several studies (Ma et al. 2024; Shao et al. 2023; Xiao et al. 2023). These PTQ methods typically employ the following quantization procedure:

$$\begin{aligned}
\mathbf{W}^{\text{Int4}} &= \text{clamp}\left(\text{round}\left(\mathbf{Temp}\right), 0, 2^N - 1\right), \\
\mathbf{Temp} &= \frac{\mathbf{W}^{\text{FP16}}}{\mathbf{Scale}^{\text{FP16}}} - \mathbf{ZeroPoint}^{\text{Int4}} \\
\mathbf{ZeroPoint}^{\text{Int4}} &= \text{round}\left(\frac{\mathbf{W}_{\text{min}}^{\text{FP16}}}{\mathbf{Scale}^{\text{FP16}}}\right), \\
\mathbf{Scale}^{\text{FP16}} &= \frac{\mathbf{W}_{\text{max}}^{\text{FP16}} - \mathbf{W}_{\text{min}}^{\text{FP16}}}{2^N - 1}.
\end{aligned} \tag{7}$$

The corresponding dequantization process is described as follows:

$$\tilde{\mathbf{W}}^{\text{FP16}} = (\mathbf{W}^{\text{Int4}} - \mathbf{ZeroPoint}^{\text{Int4}}) \odot \mathbf{Scale}^{\text{FP16}} \tag{8}$$

With the ZeroPoint no longer stored in FP16 format, these PTQ methods introduce significant challenges when attempting to apply the QA-LoRA reparameterization techniques to ZeroPoint. This necessitates converting $\text{ZeroPoint}^{\text{Int4}}$ back to FP16, and entails modifying the dequantization process from Equation 8 to Equation 2. This conversion might require additional storage space and could lead to the need for replacing efficiently optimized operators on downstream devices, which is often not optimal. LoQA circumvents these issues by supporting fine-tuning of the Scale component exclusively, making it highly compatible with various PTQ methods. Experimental evidence suggests that adjustments to the Scale alone can yield effective results in Table 3, demonstrating LoQA's adaptability and efficacy in enhancing the flexibility of PTQ frameworks.

Experiments

Main Results

Comparative evaluation of the LLaMA model on the MMLU benchmark against recent competitors. We utilized Low-Rank Quantization Adaptation (LoQA) to fine-tune LLaMA models and assessed their performance on the

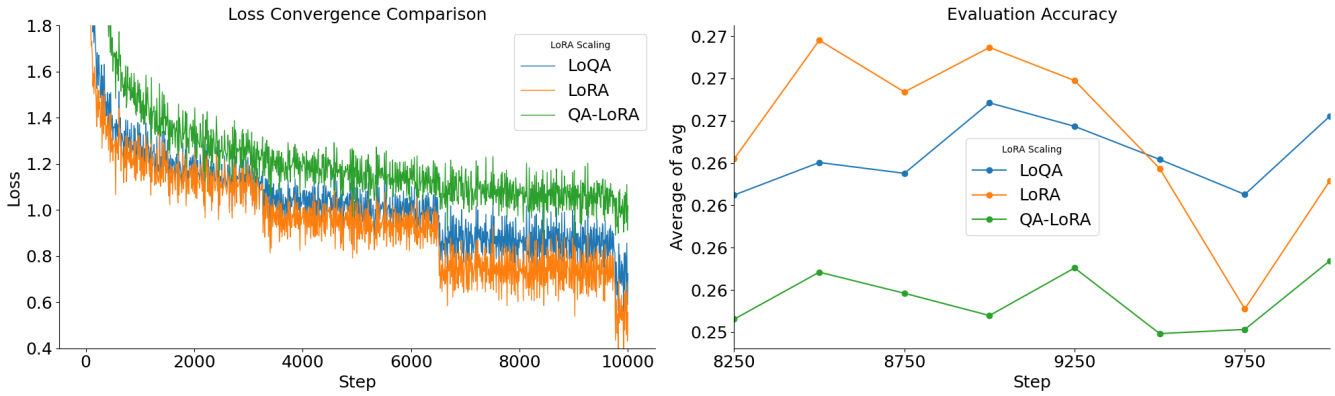

Figure 2: This figure illustrates the impact of different LoRA scaling values on LoQA-S performance. When larger scaling values are used in LoRA, LoQA-S exhibits greater instability and a higher propensity for overfitting, evidenced by sharp loss drops after each epoch and shown by the eval accuracy in the right figure.

MMLU benchmark. Table 1 summarizes results across different model sizes, fine-tuning datasets, and bit widths, comparing LoQA with similar methods like QA-LoRA (Xu et al. 2023), QLoRA (Dettmers et al. 2023), and the non-LoRA-based PEQA (Kim et al. 2023). We implemented QA-LoRA in the same environment for consistency and extracted data for other methods from published studies. Our results, especially with the Flan-v2 dataset, show consistent performance enhancements, demonstrating LoQA's broad applicability and effectiveness. For example, fine-tuning the LLaMA7B model at 3-bit width led to a 2.8% improvement in MMLU 5-shot tasks over the existing state-of-the-art, showcasing LoQA's robustness. Experiments on smaller dataset over multiple epochs indicate slight improvements over the state-of-the-art, affirming our LoRA scaling strategy's ability to prevent overfitting. **On LLaMA2 and LLaMA3 models.** We further validated the effectiveness of our method on LLaMA3. As depicted in Table 2, we fine-tuned the 7B models from LLaMA2 and the 8B models from LLaMA3, and tested them on the MMLU benchmark. Compared to the original FP16 models, the fine-tuned models demonstrated great performance. These experiments confirm that LoQA can be generalized across different families of pre-trained models.

Impact of LoRA scalings. In previous sections, we conducted experiments on the llama7b model to explore the impact of different LoRA scaling values on training loss and validation accuracy, providing insight into the relationship between LoRA scaling magnitudes and issues of underfitting and overfitting in Figure 2. The proposed LoRA scaling approach, as applied in experiments shown in Tables 1 and 3, has been tested across models of varying bit widths, demonstrating its broad applicability and effectiveness in addressing these issues.

Adjustment of Different Parts of Quantized Weights As detailed in the methods section, our innovative approach offers the flexibility to adjust either the scale alone or both the scale and zero point, expanding upon previous methodologies that primarily adjusted the zero point alone. We refer to the method that solely adjusts the scale as **LoQA-S**. In our ablation study, we investigated the individual and combined impacts of these adjustments. Applying LoQA-S to the LLaMA7B model, we evaluated performance across various bit widths and datasets. Our findings reveal that LoQA-S consistently performs well on smaller datasets. However, on larger datasets, while adjustments to the scale alone are somewhat constrained by the model's capacity and do not outperform adjustments to both scale and zero point, they still yield favorable outcomes.

Conclusion

In this paper, we introduce a novel approach named LoQA, which proposes HQ-LoRA, capable of effectively fine-tuning all quantized parameters. We also developed a novel LoRA scaling strategy called QBAS, which can adjust the scaling size based on the quantization bit-width. LoQA not only maintains memory-saving characteristics during quantization but also preserves quantization properties after fine-tuning. It can be flexibly applied to various uniform quantization methods.

Table 1: Zero-shot and five-shot accuracy percentages on the Massive Multitask Language Understanding (MMLU) dataset (Hendrycks et al. 2021). Results are grouped by the foundational model indicated in the initial row of each block. We categorize the results based on the dataset used for fine-tuning (Alpaca or Flan-v2) and the bit width utilized in quantization. The quantization bit width notation of '4+16' signifies the original QLoRA configuration, with the final inference version implemented in FP16.

Method	**Dataset**	**#Bits**	**MMLU** (0-shot)					**MMLU** (5-shot)				
			Hums.	**STEM**	**Social**	**Other**	**Avg.**	**Hums.**	**STEM**	**Social**	**Other**	**Avg.**
LLaMA-7B	–	16	32.4	26.6	31.4	37.2	32.1	33.3	29.8	37.8	38.0	34.6
QLoRA	*Alpaca*	*4+16*	*38.1*	*31.1*	*41.6*	*46.9*	*39.4*	*36.1*	*31.9*	*42.0*	*44.5*	*38.4*
QLoRA w/ GPTQ	Alpaca	4	35.7	30.9	38.0	44.0	37.1	33.8	31.3	37.4	42.2	36.0
PEQA	Alpaca	4	–	–	–	–	–	34.9	28.9	37.5	40.1	34.8
QA-LoRA	Alpaca	4	38.7	35.6	46.7	45.9	**41.5**	37.9	35.4	45.9	46.8	41.2
LoQA	Alpaca	4	39.1	34.6	46.2	46.3	41.4	39.1	34.7	47.1	47.8	**41.9**
QLoRA w/ GPTQ	Alpaca	3	31.5	28.9	31.8	36.8	32.2	31.6	30.1	35.6	39.8	34.0
QA-LoRA	Alpaca	3	33.7	32.1	39.4	41.5	**36.4**	34.6	32.6	41.7	42.7	**37.6**
LoQA	Alpaca	3	34.0	30.2	36.3	39.1	34.9	36.1	31.2	40.7	42.4	37.5
QLoRA w/ GPTQ	Alpaca	2	24.1	22.1	22.5	23.7	23.2	23.4	26.2	26.4	28.4	25.8
QA-LoRA	Alpaca	2	24.8	25.5	23.7	28.0	25.5	25.4	27.8	30.0	26.7	**27.2**
LoQA	Alpaca	2	27.2	27.3	26.7	29.1	**27.6**	26.4	26.8	25.8	28.8	26.9
QLoRA	*FLAN v2*	*4+16*	*40.9*	*32.5*	*47.8*	*49.5*	*42.6*	*41.4*	*35.0*	*49.8*	*52.0*	*44.3*
QLoRA w/ GPTQ	FLAN v2	4	39.7	32.5	46.4	48.1	41.6	36.5	33.7	46.9	50.3	41.4
QA-LoRA	FLAN v2	4	41.6	33.9	49.5	49.4	43.5	41.8	35.6	53.7	50.8	45.2
LoQA	FLAN v2	4	42.7	34.3	53.1	51.7	**45.3**	44.0	37.2	56.1	52.3	**47.1**
QLoRA w/ GPTQ	FLAN v2	3	36.7	30.2	38.4	40.1	36.5	32.2	31.7	42.7	42.8	36.9
QA-LoRA	FLAN v2	3	39.1	30.6	45.5	45.9	40.2	40.8	34.7	50.5	49.8	43.7
LoQA	FLAN v2	3	40.3	31.1	47.8	49.0	**42.0**	42.8	33.8	52.2	52.3	**45.1**
QLoRA w/ GPTQ	FLAN v2	2	24.1	22.5	22.3	23.8	23.3	23.9	25.3	26.2	25.3	25.0
QA-LoRA	FLAN v2	2	34.1	30.0	37.2	39.8	35.2	31.8	38.1	34.5	38.5	33.2
LoQA	FLAN v2	2	35.4	28.5	39.9	38.9	**35.7**	34.2	28.8	41.0	40.5	**36.0**
LLaMA-13B	–	16	40.6	36.7	48.9	48.0	43.3	44.0	35.9	53.2	52.9	46.3
QLoRA	*Alpaca*	*4+16*	*45.2*	*38.3*	*55.0*	*54.6*	*48.1*	*46.0*	*37.3*	*55.8*	*55.1*	*48.4*
QLoRA w/ GPTQ	Alpaca	4	44.7	38.0	54.4	54.0	47.6	45.4	37.4	55.7	54.3	48.0
PEQA	Alpaca	4	–	–	–	–	–	43.0	37.7	53.6	49.0	45.0
QA-LoRA	Alpaca	4	42.5	36.0	52.1	52.9	45.6	46.4	36.1	53.4	53.4	47.4
LoQA	Alpaca	4	43.5	35.5	51.1	53.3	**45.7**	45.8	36.4	54.3	54.4	**47.6**
QLoRA w/ GPTQ	Alpaca	3	43.5	36.2	52.3	52.6	45.9	43.6	36.1	53.0	52.7	46.1
QA-LoRA	Alpaca	3	42.3	36.0	51.2	51.6	45.1	42.9	36.8	52.6	51.2	45.6
LoQA	Alpaca	3	42.9	36.8	52.3	53.1	**46.0**	43.9	37.0	54.2	53.2	**46.8**
QLoRA w/ GPTQ	Alpaca	2	27.7	27.6	31.8	29.7	29.0	29.0	27.1	33.4	34.8	30.9
QA-LoRA	Alpaca	2	28.9	30.0	34.5	36.2	32.0	29.4	29.7	35.5	37.0	**32.5**
LoQA	Alpaca	2	30.9	29.3	33.8	36.2	**32.4**	29.3	28.0	34.8	38.8	32.4
QLoRA	*FLAN v2*	*4+16*	*48.0*	*39.2*	*58.2*	*56.7*	*50.3*	*49.9*	*40.1*	*60.2*	*57.9*	*51.9*
QLoRA w/ GPTQ	FLAN v2	4	47.6	39.6	57.6	56.0	50.0	49.4	40.9	59.7	57.6	51.7
QA-LoRA	FLAN v2	4	46.4	36.0	57.5	55.8	48.8	49.9	39.6	60.2	56.6	51.5
LoQA	FLAN v2	4	48.9	41.2	59.9	58.0	**51.8**	50.2	42.7	62.2	58.3	**53.1**
QLoRA w/ GPTQ	FLAN v2	3	46.6	37.9	55.9	55.7	48.9	46.5	38.2	57.2	56.1	49.3
QA-LoRA	FLAN v2	3	45.7	38.0	57.3	54.3	48.5	47.2	40.2	59.5	56.2	50.5
LoQA	FLAN v2	3	47.4	40.3	59.5	54.7	**50.2**	48.0	42.2	61.5	57.4	**51.9**
QLoRA w/ GPTQ	FLAN v2	2	36.2	30.3	40.8	44.1	37.8	36.6	32.0	43.8	44.2	38.9
QA-LoRA	FLAN v2	2	39.9	31.5	46.9	45.2	40.9	38.3	34.2	47.4	46.9	41.4
LoQA	FLAN v2	2	41.0	34.3	49.5	48.4	**43.1**	39.5	36.1	51.3	48.1	**43.3**
LLaMA-33B	–	16	51.0	42.7	63.3	60.4	54.1	56.2	45.9	67.1	63.9	58.2
QLoRA	*Alpaca*	*4+16*	*52.2*	*44.9*	*64.3*	*61.8*	*55.5*	*55.4*	*46.0*	*66.4*	*63.6*	*57.7*
QLoRA w/ GPTQ	Alpaca	4	51.7	44.7	63.4	61.0	54.9	53.9	46.6	66.3	62.9	57.1
QA-LoRA	Alpaca	4	51.6	44.9	65.0	61.8	**55.4**	55.8	46.4	67.0	64.0	**58.1**
LoQA	Alpaca	4	51.8	42.5	63.6	61.3	54.6	53.8	44.7	65.0	62.0	56.2
QLoRA	*FLAN v2*	*4+16*	*56.3*	*46.5*	*68.6*	*64.6*	*58.8*	*57.2*	*48.6*	*69.8*	*65.2*	*60.0*
QLoRA w/ GPTQ	FLAN v2	4	54.9	46.4	68.2	63.6	58.0	57.4	48.6	69.2	64.9	59.8
QA-LoRA	FLAN v2	4	53.6	44.8	66.3	62.4	56.5	55.9	47.4	69.6	65.1	59.2
LoQA	FLAN v2	4	55.5	46.3	69.2	63.6	**58.4**	58.7	49.3	71.0	65.8	**61.0**

Table 2: 0-shot and 5-shot MMLU accuracy (%) based on the LLaMA2 and LLaMA3.

Method	Data	#Bits	MMLU (0-shot) Hums. (↑)	STEM (↑)	Social (↑)	Other (↑)	Avg. (↑)	MMLU (5-shot) Hums. (↑)	STEM (↑)	Social (↑)	Other (↑)	Avg. (↑)
LLaMA2-7B	–	16	38.9	32.9	46.6	44.9	40.7	43.0	36.4	51.4	52.2	45.5
QA-LoRA	Alpaca	4	40.6	37.2	50.9	49.6	44.2	40.9	36.7	48.6	50.6	43.9
LoQA	Alpaca	4	42.2	37.3	52.3	52.0	**45.6**	42.3	38.1	52.5	53.0	**46.1**
QA-LoRA	FLAN v2	4	45.6	38.9	56.0	54.8	48.6	45.1	39.9	58.3	56.4	49.5
LoQA	FLAN v2	4	46.0	39.0	59.0	56.3	**49.7**	46.1	41.1	61.4	57.9	**51.1**
LLaMA3-8B	–	16	-	-	-	-	-	59.0	55.3	76.0	71.5	64.8
LoQA	Alpaca	4	51.0	49.7	69.8	65.6	58.2	42.7	49.7	68.5	64.2	54.8
LoQA	FLAN v2	4	53.0	49.6	69.3	65.7	58.8	44.3	50.8	70.4	66.8	56.6

Table 3: 0-shot and 5-shot accuracy (%) on the Massive Multitask Language Understanding (MMLU) dataset (Hendrycks et al. 2021). Each block is based on the same foundation model specified at the first row. We organize all results using the fine-tuning dataset (Alpaca or Flan-v2) and the bit width of quantization. The bit width of '$4+16$' refers to the original QLoRA where the final version for inference is in FP16.

Method	Dataset	#Bits	MMLU (0-shot) Hums.	STEM	Social	Other	Avg.	MMLU (5-shot) Hums.	STEM	Social	Other	Avg.
LLaMA-7B	–	16	32.4	26.6	31.4	37.2	32.1	33.3	29.8	37.8	38.0	34.6
QLoRA	*Alpaca*	*4+16*	*38.1*	*31.1*	*41.6*	*46.9*	*39.4*	*36.1*	*31.9*	*42.0*	*44.5*	*38.4*
QA-LoRA	Alpaca	4	38.7	35.6	46.7	45.9	41.5	37.9	35.4	45.9	46.8	41.2
LoQA-S	Alpaca	4	39.4	34.3	47.5	46.9	**41.8**	38.2	34.3	46.5	46.9	41.2
LoQA	Alpaca	4	39.1	34.6	46.2	46.3	41.4	39.1	34.7	47.1	47.8	**41.9**
QA-LoRA	Alpaca	3	33.7	32.1	39.4	41.5	**36.4**	34.6	32.6	41.7	42.7	37.6
LoQA-S	Alpaca	3	33.9	31.4	37.5	39.9	35.5	35.9	32.7	40.8	43.7	**38.1**
LoQA	Alpaca	3	34.0	30.2	36.3	39.1	34.9	36.1	31.2	40.7	42.4	37.5
QA-LoRA	Alpaca	2	24.8	25.5	23.7	28.0	25.5	25.4	27.8	30.0	26.7	**27.2**
LoQA-S	Alpaca	2	26.5	25.9	24.9	28.2	26.4	27.2	25.6	26.5	27.0	26.7
LoQA	Alpaca	2	27.2	27.3	26.7	29.1	**27.6**	26.4	26.8	25.8	28.8	26.9
QLoRA	*FLAN v2*	*4+16*	*40.9*	*32.5*	*47.8*	*49.5*	*42.6*	*41.4*	*35.0*	*49.8*	*52.0*	*44.3*
QA-LoRA	FLAN v2	4	41.6	33.9	49.5	49.4	43.5	41.8	35.6	53.7	50.8	45.2
LoQA-S	FLAN v2	4	42.6	34.9	53.1	51.0	45.2	42.7	36.7	55.3	51.4	46.2
LoQA	FLAN v2	4	42.7	34.3	53.1	51.7	**45.3**	44.0	37.2	56.1	52.3	**47.1**
QA-LoRA	FLAN v2	3	39.1	30.6	45.5	45.9	40.2	40.8	34.7	50.5	49.8	43.7
LoQA-S	FLAN v2	3	41.0	33.8	51.1	49.4	**43.6**	42.8	36.1	52.6	50.4	**45.3**
LoQA	FLAN v2	3	41.3	33.4	50.7	49.4	43.5	41.5	36.0	53.7	50.4	45.0
QA-LoRA	FLAN v2	2	34.1	30.0	37.2	39.8	35.2	31.8	38.1	34.5	38.5	33.2
LoQA-S	FLAN v2	2	35.3	29.7	39.9	41.4	**36.5**	32.4	31.2	39.2	39.2	35.2
LoQA	FLAN v2	2	35.4	28.5	39.9	38.9	35.7	34.2	28.8	41.0	40.5	**36.0**

References

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models Are Few-Shot Learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. LLM.int8(): 8-Bit Matrix Multiplication for Transformers at Scale. In *Advances in Neural Information Processing Systems*.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2024. QLoRA: Efficient Finetuning of Quantized LLMs. *Advances in Neural Information Processing Systems*, 36.

Dettmers, T.; Svirschevski, R.; Egiazarian, V.; Kuznedelev, D.; Frantar, E.; Ashkboos, S.; Borzunov, A.; Hoefler, T.; and Alistarh, D. 2023. SpQR: A Sparse-Quantized Representation for Near-Lossless LLM Weight Compression. *arXiv preprint arXiv:2306.03078*.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022. GPTQ: Accurate Post-Training Quantization for Generative Pre-Trained Transformers. *arXiv preprint arXiv:2210.17323*.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2023. GPTQ: Accurate Post-Training Compression for Generative Pretrained Transformers. In *International Conference on Learning Representations*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.

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*.

Kim, J.; Lee, J. H.; Kim, S.; Park, J.; Yoo, K. M.; Kwon, S. J.; and Lee, D. 2023. Memory-Efficient Fine-Tuning of Compressed Large Language Models via Sub-4-Bit Integer Quantization. *arXiv preprint arXiv:2305.14152*.

Köksal, A.; Schick, T.; Korhonen, A.; and Schütze, H. 2023. Longform: Optimizing Instruction Tuning for Long Text Generation with Corpus Extraction. *arXiv preprint arXiv:2304.08460*.

Kopiczko, D. J.; Blankevoort, T.; and Asano, Y. M. 2023. Vera: Vector-Based Random Matrix Adaptation. *arXiv preprint arXiv:2310.11454*.

Le Scao, T.; Fan, A.; Akiki, C.; Pavlick, E.; Ilić, S.; Hesslow, D.; Castagné, R.; Luccioni, A. S.; Yvon, F.; Gallé, M.; et al. 2023. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. *arXiv preprint arXiv:2211.05100*.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Dang, X.; and Han, S. 2023. AWQ: Activation-Aware Weight Quantization for LLM Compression and Acceleration. *arXiv preprint arXiv:2306.00978*.

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*.

Ma, Y.; Li, H.; Zheng, X.; Ling, F.; Xiao, X.; Wang, R.; Wen, S.; Chao, F.; and Ji, R. 2024. AffineQuant: Affine Transformation Quantization for Large Language Models. *arXiv preprint arXiv:2403.12544*.

Meng, F.; Wang, Z.; and Zhang, M. 2024. PiSSA: Principal Singular Values and Singular Vectors Adaptation of Large Language Models. *arXiv preprint arXiv:2404.02948*.

Shao, W.; Chen, M.; Zhang, Z.; Xu, P.; Zhao, L.; Li, Z.; Zhang, K.; Gao, P.; Qiao, Y.; and Luo, P. 2023. OmniQuant: Omnidirectionally Calibrated Quantization for Large Language Models. *arXiv preprint arXiv:2308.13137*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023a. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. LLaMA 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. In *International Conference on Machine Learning*.

Xu, Y.; Xie, L.; Gu, X.; Chen, X.; Chang, H.; Zhang, H.; Chen, Z.; Zhang, X.; and Tian, Q. 2023. QA-LoRA: Quantization-Aware Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2309.14717*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. OPT: Open Pre-Trained Transformer Language Models. *arXiv preprint arXiv:2205.01068*.

Zhu, X.; Li, J.; Liu, Y.; Ma, C.; and Wang, W. 2023. A Survey on Model Compression for Large Language Models. *arXiv preprint arXiv:2308.07633*.