# Exploring Post-training Quantization in LLMs from Comprehensive Study to Low Rank Compensation

Zhewei Yao*[1], Xiaoxia Wu*[1], Cheng Li[1], Stephen Youn[1], Yuxiong He[1]

[1] DeepSpeed of Microsoft

## Abstract

Post-training quantization (PTQ) has emerged as a promising technique for mitigating memory consumption and computational costs in large language models (LLMs). However, a systematic examination of various quantization schemes, model families, and quantization bit precision has been absent from the literature. In this paper, we conduct a comprehensive analysis of these factors by investigating the effects of PTQ on weight-only, activation-only, and weight-and-activation quantization using diverse methods such as round-to-nearest (RTN), GPTQ, ZeroQuant, and their variants. We apply these methods to two distinct model families with parameters ranging from 125M to 176B. Our contributions include: (1) a sensitivity analysis revealing that activation quantization is generally more susceptible to weight quantization, with smaller models often outperforming larger models in terms of activation quantization; (2) an evaluation and comparison of existing PTQ methods to optimize model size reduction while minimizing the impact on accuracy, revealing that none of the current methods can achieve the original model quality for quantization with either INT4-weight or INT4-weight-and-INT8-activation; (3) based on these insights, we propose an optimized method called Low-Rank Compensation (LoRC), which employs low-rank matrices to enhance model quality recovery with a minimal increase in model size.

## Introduction

Large language models (LLMs) like Codex (Copilot 2021) and ChatGPT (OpenAI 2022) have demonstrated breakthrough performance across various benchmarks, such as natural language understanding and generation, and are now integrated into everyday applications. However, efficiently serving LLMs has become a pressing concern due to their significant memory consumption and computational demands. Unlike classification or diffusion models, LLMs present unique challenges, as they involve two distinct phases: prompt and generation. The prompt phase is primarily compute-bound, while the generation phase, with low batch size and KV cache, is mainly memory-bound (Pope et al. 2022).

As the progression of hardware bandwidth lags behind that of computational demand (Gholami et al. 2021), the resource demands of extra-large models such as MT-NLG-530B (Smith et al. 2022)—which necessitates the deployment of multiple nodes for operation—escalate, adding to the complexities of cross-node communication. This has emphasized the urgency to curtail both the size and computational expense of Large Language Models (LLMs). An increasingly effective solution to these issues is post-training quantization (PTQ). This method aids in the reduction of training prerequisites while simultaneously lowering the bit precision of weights and activations to either INT4 or INT8.

While the effectiveness of post-training quantization (PTQ) has been underscored in a number of recent studies (Yao et al. 2022; Frantar et al. 2022; Xiao et al. 2022; Dettmers and Zettlemoyer 2022), a comprehensive, systematic investigation into several key dimensions of this technique remains to

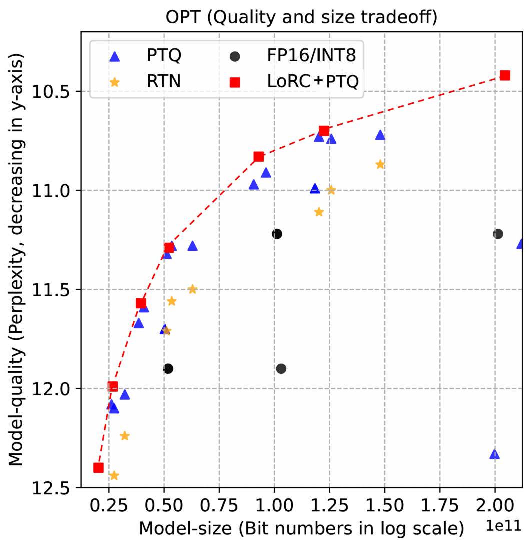

Figure 1: The model size and quality trade-off of different quantization methods on models from OPT families. Here PTQ (with fine-grained quantization) represents the method from (Yao et al. 2022; Frantar et al. 2022), RTN means the naive round-to-nearest baseline (with fine-grained quantization as well), and FP16/INT8 is used as the no-accuracy-loss baseline. LoRC is our proposed method that works seamless with PTQ. Note that we drop all diverged points for better visualization.

*Equal Contribution.

be undertaken. Specifically, the extant literature falls short in providing thorough coverage of the functionality of various PTQ methods or the sensitivity of disparate models. Moreover, despite current quantization methods demonstrating promising results in the reduction of model sizes, the question persists as to whether these methods are achieving their optimal potential in minimizing Large Language Models (LLMs) sizes.

With these observations in mind, our study sets forth to address two salient questions: (1) When subjected to quantization, do LLMs of varying sizes and pretraining data exhibit similar behavior? (2) Are existing quantization methods truly leveraging their full potential in reducing the sizes of LLMs?

**Contribution.** To elucidate these queries, we undertake an exhaustive examination of the impact of PTQ on weight-only, activation-only, and combined weight-and-activation quantization. This investigation incorporates a range of PTQ methods, including round-to-nearest (RTN), GPTQ (Frantar et al. 2022), ZeroQuant (Yao et al. 2022), and their respective variants. To broaden the scope of our analysis, we focus on two distinct model families, OPT (Zhang et al. 2022) and BLOOM (Scao et al. 2022), spanning model sizes from 125M to a massive 176B. Our code will be made available for reproduction. In summary, we make the following contributions:

(1) We provide a thorough **sensitivity analysis** to demonstrate that a) Activation quantization is generally more sensitive to weight quantization; Smaller models usually have better activation quantization performance than the relative larger model. b) Different model families show different INT8 activation quantization behaviors; Particularly for large models, BLOOM-176B has small accuracy drops (about 1 perplexity or PPL) but OPT-30B and -66B experience worse performance.

(2) We carry out a detailed evaluation and comparison of current PTQ methods, utilizing optimal configurations to maximize model size reduction while minimizing accuracy impact. We found that the current existing method can barely achieve less than 0.1 PPL points degradation for quantization with either INT4-weight or INT4-weight-and-INT8-activation (W4A8). To recover the 0.1 PPL, we strive to push the boundaries of employing **fine-grained quantization** (FGQ) techniques. We observe FGQ is able to recovered points degradation of <0.1 PPL for large models (>13B) for INT4 weight quantization, but there are still non-negligible model quality drops.

(3) Based on the above understanding, we further optimize existing methods and introduce a technique called **Lo**w **R**ank **C**ompensation (LoRC), which employs low-rank matrix factorization on the quantization error matrix. Complementary to FGQ, LoRC plays a crucial role in enhancing the full model quality recovery, while there is little increase of the model size.

In Figure 1, we provide model size and quality trade-offs for both OPT families. As can be seen, using LoRC on top of PTQ methods from (Yao et al. 2022; Frantar et al. 2022) and fine-grained quantization, we set a new quantization Pareto frontier for LLMs. Meanwhile, we recommend the following setting for quantizing LLMs with LoRC (Note that activation quantization should be only applied if necessary): (1) For larger models (>10B), fine-grained (block size 64–256) 4-bit weight quantization plus 8-bit activation quantization (block size 64–256) with PTQ can be used for real deployment; (2) For middle-size models (<10B and >1B), per-row INT8 quantization plus fine-grained (block size 64–256) INT8 activation quantization can be used with PTQ from (Frantar et al. 2022; Yao et al. 2022); (3) For smaller models (<1B), per-row W8A8 (INT8 weight and INT8 activation) RTN is enough based on (Yao et al. 2022). [1]

## Related Work

Different quantization methods (Shen et al. 2020; Zafrir et al. 2019; Fan et al. 2020; Zhang et al. 2020; Bai et al. 2020; Esser et al. 2019; Tao et al. 2022; Kim et al. 2021) for transformer-based models (Vaswani et al. 2017) have been explored for a while. However, most of those works need quantization-aware finetuning or even expensive quantization-aware knowledge distillation (Hinton, Vinyals, and Dean 2014). Due to the cost of training/finetuning LLMs (Polino, Pascanu, and Alistarh 2018; Jiao et al. 2019; Tao et al. 2022; Wu et al. 2022, 2023), it is a challenge for practitioners/researchers to do finetuning/distillation on those LLMs, particularly for models like GPT-3-175B (Brown et al. 2020) and BLOOM-176B (Scao et al. 2022).

Post-training quantization (PTQ) (Zadeh et al. 2020; Bondarenko, Nagel, and Blankevoort 2021) is an alternative way to quantize the model with no/minimal finetuning requirement. Along this line, several recent works focus on LLMs (beyond the million-parameter scale). (Yao et al. 2022) proposes vector-based INT8 quantization with layer-by-layer knowledge distillation to overcome the training cost and quantization error introduced by LLMs. (Dettmers et al. 2022) uses similar vector-based INT8 quantization weight plus mixed-precision (INT8/FP16) quantization for activation to overcome the sensitivity of activation quantization. However, the inference speed of (Dettmers et al. 2022) is generally even slower than FP16 baseline (Big-Science 2022) due to the difficulty of implementing mixed-precision calculation within a single tensor. More recently, (Frantar et al. 2022) extends OBQ (Frantar and Alistarh 2022; Hassibi and Stork 1993; LeCun, Denker, and Solla 1990) on LLMs for INT4 weight-only quantization and shows great efficiency on quantization and latency, and (Xiao et al. 2022) shows the outliers from activations can be smoothed out by migrating the quantization difficulty from activations to its associated weights. However, (Xiao et al. 2022) can only work for W8A8 quantization as lower weight precision (INT4) itself already leads to significant accuracy degradation, and the accuracy drop is larger than 0.1 PPL points, which as discussed in the later section is sub-optimal. (Dettmers and Zettlemoyer 2022) shows the scaling law of weight-only quantization with the simplest round-to-nearest baseline, but it does not consider the weight-and-activation quantization and/or the above PTQ optimization methods. As can be seen from Figure 1, by using PTQ optimization methods, the model quality can be significantly improved.

[1]Full version is in arXiv: 2303.08302.

| Class | ***Class***-1 | ***Class***-2 | ***Class***-3 |
|---|---|---|---|
| PPL Degradation | $\leq$0.1 | >0.1 & $\leq$0.5 | >0.5 |

Table 1: Classification of quantization sensitivity (or quantization loss). The sensitivity increases from Class-1 to Class-3.

Different than existing works, our paper extensively tests the effect of (1) different quantization schemes, e.g., symmetric and asymmetric quantization, (2) different PTQ methods, e.g., (Yao et al. 2022; Frantar et al. 2022), (3) different model families, e.g., (Scao et al. 2022; Zhang et al. 2022), (4) different quantization coverage, e.g., weight-only and weight-and-activation quantization, and (5) other discussions, e.g., the effect of quantization granularity. As such, we provide a much more comprehensive understanding of post-training quantization for large language models compared to the previous works.

## Would Different Model Families Behave Similarly On Quantization?

There are mainly two categories of PTQ for LLMs, i.e., weight-only quantization (Frantar et al. 2022) and weight-and-activation quantization (Dettmers et al. 2022; Yao et al. 2022; Xiao et al. 2022). In the latter, it is uniformly observed across all studies that activation quantization demonstrates greater sensitivity than weight quantization. However, prior research tends to concentrate on a single (family) model to emphasize the necessity of their proposed quantization technique. A comprehensive and systematic evaluation of this PTQ methodology, particularly the sensitivity of weight/activation quantization for varying model sizes and distinct model families, has yet to be undertaken. Hence, we conduct an examination on both the OPT (Zhang et al. 2022) and BLOOM (Scao et al. 2022) families to elucidate the quantization sensitivity of weight and activation.

**Sensitivity Setting.** We use the zero-shot validation perplexity (PPL) differential on three datasets, namely, Wikitext-2 (Merity et al. 2017), PTB (Marcinkiewicz 1994), and C4 (Raffel et al. 2019), before and after the quantization of these LLMs to illustrate their sensitivity, as PPL is significantly correlated to zero-shot/few-shot accuracy measurement (Dettmers and Zettlemoyer 2022). Specifically, a higher PPL drop indicates enhanced quantization sensitivity. For simplicity, we also categorize quantization sensitivity (or quantization loss) into three different classes as depicted in Table 1. Notably, the threshold is chosen because when the model size approximately doubles (e.g., 13B vs. 30B, and 30B vs. 66B), the PPL improvement is about 0.5 (see Table 2). The sensitivity (or loss) incrementally increases as the class number ascends. From a practical standpoint, we favor lower quantization sensitivity (accuracy loss), making ***Class***-1 the optimal-loss post-training quantization.

Both symmetric and asymmetric quantization are used to gauge the quantization sensitivity and the advantage of asymmetric quantization is highlighted. Particularly, we implement per-row quantization (Frantar et al. 2022) for weight and per-token quantization for activation (Yao et al. 2022).

**Robustness of Weight-only Quantization for Large Models.** The results of weight-only quantization in OPT and BLOOM models are summarized in Table 2. INT8 weight-only quantization, either symmetric or asymmetric, results in negligible accuracy loss (less than 0.05, i.e., ***Class***-1). Consequently, for tasks oriented towards generation, FP16 weight can simply be replaced with INT8 weight to reduce memory usage. For INT4 quantization, the asymmetric method outperforms the symmetric approach in accuracy, attributable to its superior utilization of the quantization range. Interestingly, larger models exhibit better tolerance to low-precision quantization (i.e., INT4) than smaller models, with a few exceptions such as OPT-66B.[2] Particularly, BLOOM-176B shows PPL degradation (around 0.3 points) in ***Class***-2, which could explain why the large GLM-130B (Zeng et al. 2022) can operate with INT4 weight-only quantization out of the box with acceptable accuracy impact.

**Challenge Encountered in Activation Quantization for Large Models.** Activation quantization has consistently proven more difficult than weight quantization (Yao et al. 2022; Dettmers et al. 2022), as illustrated in Table 2. When compared to weight-only quantization, activation-only quantization indicates that asymmetric quantization can significantly improved performance over symmetric quantization. Moreover, contrary to weight-only quantization, smaller models typically exhibit better tolerance to activation quantization, as their hidden dimension is smaller and the activation dynamic range is also narrower than larger models (Yao et al. 2022). It should be noted that for models larger than 10B, all fall into ***Class***-3, indicating a degradation of more than 0.5 PPL points.

The last two rows of Table 2 show that different model families exhibit significantly different behaviors. BLOOM does not exhibit divergence issues even up to a model size of 176B, whereas OPT displays very poor performance from a model size of 6.7B (larger models with INT8 activation have even worse PPL). This could again be attributed to the Layer Norm issue within the OPT-family[2].

**Findings 1 on Sensitivity Analysis. (1)** INT8 weight-only quantization can serve as a standard method for reducing memory costs in LLMs, with negligible degradation in accuracy. **(2)** INT4 weight-only quantization for small models results in substantial accuracy degradation (***Class***-3), but this effect lessens as the model size increases (***Class***-2). **(3)** Contrary to (2), INT8 activation results in minimal accuracy drops for small models (***Class***-1) but larger models exhibit greater drops (***Class***-3). **(4)** With INT8 activation, BLOOM shows no divergence issues up to a model size of 176B, whereas OPT performs poorly from $\geq$ 6.7B model sizes.

[2](Frantar et al. 2022) discovered that OPT-66B has a high proportion of dead neurons in the early layers, which might influence the compression capability. We also identify another potential reason: the Layer Norm of the OPT-family is not well trained (except OPT-350M), with the weight and the bias being all 1's and 0's.

| Precision | OPT-6.7b | OPT-13b | OPT-30b | OPT-66b | BLM-1.7b | BLM-3b | BLM-7.1b | BLM-176b |
|---|---|---|---|---|---|---|---|---|
| W16-A16 | 11.90 | 11.22 | 10.70 | 10.33 | 20.43 | 17.58 | 14.96 | 10.90 |
| W8$^{\text{sym}}$-A16 | 11.90 | 11.22 | 10.70 | 10.33 | 20.43 | 17.59 | 14.97 | 10.90 |
| W8$^{\text{asym}}$-A16 | 11.90 | 11.22 | 10.70 | 10.33 | 20.45 | 17.59 | 14.97 | 10.90 |
| W4$^{\text{sym}}$-A16 | 14.36 | 12.73 | 11.77 | 97.05 | 23.18 | 19.36 | 16.27 | 11.28 |
| W4$^{\text{asym}}$-A16 | 13.44 | 12.09 | 11.52 | 31.52 | 22.47 | 19.01 | 15.90 | 11.20 |
| W16-A8$^{\text{sym}}$ | 26.04 | 3171.49 | 2048.21 | 2638.09 | 20.68 | 17.73 | 15.28 | 12.10 |
| W16-A8$^{\text{asym}}$ | 12.62 | 15.36 | 23.57 | 561.35 | 20.52 | 17.65 | 15.14 | 11.62 |

Table 2: Average PPL of OPT and BLOOM (BLM).

## Are Existing Quantization Methods Optimally Harnessing the Potential to Minimize Sizes?

Numerous lightweight optimization-based methods have been proposed, which update the model weights during quantization. These methods such as (Yao et al. 2022; Frantar et al. 2022; Xiao et al. 2022), unlike quantization-aware training, only require a small portion of the training data and a limited training time. Particularly, GPTQ (Frantar et al. 2022) and ZeroQuant (Yao et al. 2022), have proven to be effective and efficient in terms of GPU resources, time cost, and data usage for INT4 weight quantization.[3] In this work, we focus on the variants of GPTQ and ZeroQuant as well as the most straightforward baseline, round-to-nearest neighborhood (RTN).

**RTN** directly applies PTQ on the trained data to perform the quantization. Specifically, for symmetric quantization, we set $S = max(abs(x))$ and $Z = 0$; for asymmetric quantization, we set $S = max(x) - min(x)$ and $Z = min(x)$.

**GPTQ** extends the OBQ (Frantar and Alistarh 2022). It tries to optimize the following non-linear least square problem, $\min_{\hat{W}} \|Wx - \hat{W}x\|_2^2$ where $W$ is the weight, $x$ is the activation, and $\hat{W}$ is a quantized weight. GPTQ employs second-order methods to obtain a closed-form solution. In addition, the quantization for each weight matrix is performed column-/row-wisely and the quantization errors from previous columns will be passed to those columns not yet quantized. See(Frantar and Alistarh 2022; Frantar et al. 2022) for more details.

**ZQ-Global** is the original method proposed in (Yao et al. 2022), where authors treat each layer as a small neural network (a.k.a., subnetwork) and use the FP16 subnetwork as the teacher model to distill the quantized one with a few hundred iterations, i.e., $\min_{\hat{\theta}} \|f_\theta(x) - f_{\hat{\theta}}(x)\|_2^2$, where $\theta$ is a set of weights, $\hat{\theta}$ is the quantized version, $f\theta$ is the subnetwork with parameters $\theta$, and $x$ is the input. Thus, it can significantly reduce the GPU resource requirement and time cost.

**ZQ-Local** is an extension mode of ZQ-Global for further GPU requirement reduction and training cost reduction. Particularly, instead of using each transformer layer as the subnetwork, we treat each linear layer as the subnetwork. This method can be viewed as an iterative first-order optimization method (e.g., SGD) to solve $\min_{\hat{W}} \|Wx - \hat{W}x\|_2^2$.

**Experimental Setup.** We compare the four methods mentioned above on weight-only and weight-and-activation quantization. As weight quantization is always static (i.e., it does not change during inference), there is virtually no system performance difference between symmetric and asymmetric quantization.[4] We use asymmetric quantization for better accuracy, and the conclusions would hold similarly for symmetric quantization. For parameters used for GPTQ, ZQ-Local, and ZQ-Global, please refer to Appendix in our arxiv paper. An interesting finding for ZeroQuant is that the hyperparameters (e.g., learning rate and its scheduler) provided in the original work (Yao et al. 2022) are sub-optimal. In this work, we find the best configurations for ZQ-Local and ZQ-Global and denote them as ZQ-Local$^*$ and ZQ-Global$^*$, respectively, with the best tuned results. To ensure consistent and comparable results, we set a fixed random seed for our experiments. In the context of post-training quantization, varying the random seed has minimal impact on the final results.

**Evaluation of Weight-only Quantization.** The results from weight-only quantization using OPT and Bloom are presented in Table 3. The findings indicate that the larger models tend to be less sensitive to INT4 weight-only quantization. This observation holds true across all methods (RTN, GPTQ, ZQ-Local$^*$, and ZQ-Global$^*$) with the exception of OPT-66B, which shows greater degradation than OPT-30B. It is noteworthy that light-weight optimization-based methods significantly outperform the RTN baseline in terms of accuracy. For instance, these methods substantially reduce the degradation in perplexity of OPT-30B/66B compared to baseline. Most quantized models with parameters greater than 6.7B fall under Class II, indicating their potential for real-world applications. For instance, the quality of INT4 OPT-30B (66B) is superior to that of INT8 OPT-13B (30B).

Among the optimization-based methods, ZQ-Global$^*$ gen-

[3]We tested the method proposed by (Xiao et al. 2022) but did not find it better than others for INT4 weight quantization.

[4]The bias term (a.k.a., the zero point) can be simply fused into the previous activation quantization kernel (Yao et al. 2022).

erally performs better on smaller models (those with fewer than 1B parameters), while GPTQ excels on larger models. ZQ-Local$^*$ does not outperform GPTQ or ZQ-Global$^*$-— a reasonable outcome given that GPTQ employs a closed-form solution to solve the non-linear quadratic problem and ZQ-Global$^*$ optimizes a larger subnetwork. The inferior performance of ZQ-Global$^*$ compared to GPTQ for larger models is unexpected since ZQ-Global$^*$ optimizes an entire transformer layer while GPTQ only optimizes a single linear layer. An explanation is that larger models are more sensitive to weight updates, necessitating more advanced fine-tuning methods.

**Evaluation of Weight and Activation Quantization.** The evaluation results for existing methods using W4A8 quantization are presented in Table 3. The three light-weight optimization-based methods outperform RTN significantly, underscoring their efficacy. However, all of the results fall into either ***Class***-2 or ***Class***-3. This suggests that for certain applications, it might be more beneficial to use smaller models with fewer parameters rather than larger, quantized models.

Among quantization-based methods, ZQ-Global$^*$ and ZQ-Local$^*$ generally outperform GPTQ, which is anticipated given that GPTQ was originally designed for weight-only quantization. ZQ-Global$^*$ performs better than ZQ-Local$^*$ in most cases except for the two largest models, OPT-66B and Bloom-176B, despite having larger trainable parameters in one step. This again signifies the need for a more suitable and advanced optimization method for large language models.

**Finding 2 on Comparisons. (1)** GPTQ typically performs better for weight-only quantization, while ZeroQuant (including both ZQ-Global$^*$ and ZQ-Local$^*$) yields superior results for weight and activation quantization. **(2)** The tested optimization-based methods cannot achieve ***Class***-1 quantization error for either INT4 weight-only or W4A8 quantization with the exception of GPTQ on OPT-30B with W4A16.

## Fine-grained Quantization and Its Evaluation

With PTQ and row-wise quantization, achieving ***Class***-1 quantization error is challenging for both weight-only and weight-and-activation quantization. Generally, utilizing a smaller model with INT8 weight is more advantageous than employing a model that is twice as large with INT4 weight.

One potential solution to this issue is the implementation of finer-grained quantization schemes (Darvish Rouhani et al. 2020), where every $k$ elements possess their own scaling factor and/or zero point. This approach can significantly reduce quantization error. In the extreme case, where every single element has its own scaling factor, the original FP16 number can be precisely recovered. Importantly, block-k quantization can be implemented on modern GPUs, one of the most prevalent deep learning architectures, since the compute unit (streaming multiprocessor) of GPUs processes tiles of data (e.g., 128 by 128 tiling size) for matrix computation.

Although fine-grained quantization can substantially narrow the gap between the quantized tensor and its floating-point counterpart, the application of RTN still results in a non-trivial accuracy gap. Consequently, we build upon fine-grained quantization by employing existing optimization-based methods to further enhance accuracy. Specifically, we utilize GPTQ and ZQ-Global for all models and settings and apply ZQ-Local to OPT-66B and Bloom-176B. For the hyperparameters used in ZQ-Global and ZQ-Local, we select the top three identified in Section  for all models, except for Bloom-176B, for which we only use the top-performing hyperparameter to reduce training costs.

**4-bit Weight Quantization.** We hereby present the W4A16 results for OPT and BLOOM, as delineated in Table 4, corresponding to an array of quantization block sizes. The performance sees a significant improvement with smaller block sizes compared to per-row quantization. The point of diminishing returns, however, varies for different model sizes. For example, smaller models (such as OPT-6.7B and BLOOM-1.7b) continue to see substantial gains until the block size reduces to 32. In contrast, for larger models (those exceeding 10B, with OPT-66B as the exception), the benefits derived from smaller block sizes wane rapidly around block-256/512. Most crucially, for models equal to or larger than 13B, a smaller quantization block size results in quantization error being classified under ***Class***-1, indicating virtually negligible degradation in accuracy.

**Activation Quantization (W4A8).** To comprehend the benefits of fine-grained quantization on activation, we analyze the quantization between per-row and a block size of 128, with INT4 weight, as highlighted in Table 3. For models of considerable size, specifically those equal to or exceeding 1B, the application of such fine-grained activation quantization (Case-1) results in a substantial reduction in quantization error compared to per-row activation (Case-2). By implementing fine-grained activation quantization with weight quantization (Case-3), we are able to almost restore the performance to the level of their W4A16 counterparts.

Furthermore, we detail the impacts of varying activation quantization block sizes in Table 5 on BLOOM-176B, with INT4 weight. A trend of superior accuracy is observed with smaller block sizes in contrast to larger ones. However, the enhancement in performance reaches a saturation point when the size smaller or equal to 256, which corresponds to the range of values INT8 can represent. Despite INT8's capability to signify 256 distinct values, activation quantization errors persist due to the application of uniform quantization.

**Finding 3 on FGQ. (1)** Larger models ($\geq$10B) are capable of attaining ***Class***-1 error for 4-bit quantization. These models can leverage low-precision quantization as the model size with INT4 is similar to an INT8 model that is half its size, with improved accuracy. On the other hand, smaller models ($\leq$10B) typically reach only ***Class***-2 or ***Class***-3 error levels. **(2)** For larger models (>10B), the difference between fine-grained weight-and-activation quantization and fine-grained weight-only quantization is insignificant. **(3)** The advantage of fine-grained activation quantization fades when the block size reaches 256.

| Precision | Method | OPT-6.7b | OPT-13b | OPT-30b | OPT-66b | BLM-1.7b | BLM-3b | BLM-7.1b | BLM-176b |
|---|---|---|---|---|---|---|---|---|---|
| W16A16 | | 11.90 | 11.22 | 10.70 | 10.33 | 20.43 | 17.58 | 14.96 | 10.90 |
| **W4A16** | RTN | 13.44 | 12.09 | 11.52 | 31.52 | 22.47 | 19.01 | 15.90 | 11.20 |
| | GPTQ | 12.28 | 11.42 | 10.78 | 10.52 | 21.58 | 18.33 | 15.50 | 11.02 |
| | ZQ-Local* | 12.46 | 11.64 | 11.05 | 10.79 | 21.70 | 18.50 | 15.55 | 11.11 |
| | ZQ-Global* | 12.38 | 11.62 | 11.04 | 10.68 | 21.38 | 18.33 | 15.52 | 11.05 |
| **W4A8** | RTN | 14.80 | 26.36 | 86.26 | 815.00 | 22.75 | 19.17 | 16.19 | 12.22 |
| | GPTQ | 13.88 | 17.28 | 20.71 | 648.69 | 21.71 | 18.44 | 15.75 | 11.86 |
| | ZQ-Local* | 13.24 | 14.23 | 18.53 | 16.32 | 21.86 | 18.66 | 15.75 | 11.19 |
| | ZQ-Global* | 13.17 | 13.07 | 14.65 | 37.82 | 21.43 | 18.39 | 15.58 | 11.49 |

Table 3: Different PTQ methods on OPT and BLOOM (BLM) with asymmmetric quantization on weight or (and) activation.

| Block-size | OPT-6.7b | OPT-13b | OPT-30b |
|---|---|---|---|
| W16A16 | 11.90 | 11.22 | 10.70 |
| 1024 | 12.16 | 11.36 | 10.75 |
| 512 | 12.08 | 11.32 | 10.73 |
| 256 | 12.05 | 11.28 | 10.74 |
| 128 | 12.10 | 11.28 | 10.74 |
| 32 | 12.03 | 11.28 | 10.72 |

Table 4: W4$^{\text{asym}}$-A16 quantization out of the best result from optimization-based methods on OPT and BLOOM. N/A means that the block size is not divisible by the hidden size.

| A8 Block Size | 1024 | 512 | 256 | 128 | 32 |
|---|---|---|---|---|---|
| PPL | 10.98 | 10.97 | 10.95 | 10.95 | 10.95 |

Table 5: BLOOM-176B with different quantization block sizes on activation. Here weight is asymmetrically quantized with block size 128.

## Proposed Method to Further Push the Limit of Post-training Quantization

Building on the investigation and conclusions drawn from previous sections, it has become apparent that there is still a need for an advanced methodology to further refine the existing methods, with the objective of fully realizing the original fp16 PPL quality. In this section, we introduce a simple yet effective method called **LoRC** (Low Rank Compensation) to optimize the current existing quantization error and further bridge the gap between the quality of the original model and its quantized counterparts. **LoRC** is inspired by the employment of low-rank matrix factorization on the quantization error matrix $E := W - \hat{W}$, where $W$ represents the original weight and $\hat{W}$ is the quantized weight. LoRC approximates the error $E$ with $\hat{E} = \hat{U}\hat{V}$ by two low-rank $\hat{U}$ and $\hat{V}$:

**Step I:** Implement Singular Value Decomposition (SVD) on the error matrix $E = U\Sigma V$, where $U \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ and $V \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$ are unitary matrices, and $\Sigma \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ is a diagonal matrix with its diagonal elements ordered in a descending manner.

**Step II:** Let $\hat{E} = \hat{U}\hat{V}$ where $\hat{U} = U_m(\Sigma_m)^{\frac{1}{2}}$ and $\hat{V} = (\Sigma_m)^{\frac{1}{2}} V_m$. Here, $U_m = U_{:,1:m} \in \mathbb{R}^{d_{\text{in}} \times m}$, $V_m = V_{1:m,:} \in \mathbb{R}^{m \times d_{\text{out}}}$, and $\Sigma_m = \Sigma_{1:m,1:m} \in \mathbb{R}^{m \times m}$.

*The parameters of LoRC added to the existing model.* Consider a matrix represented as $W + UV$: where $W$ has dimensions $d \times d$, and the two low-rank matrix $U$ and $V$ are respectively of dimensions: $d \times r$ and $r \times d$. The additional parameter ratio incorporated into the existing matrix is $\mathbf{2r/d}$. For models in the OPT family with sizes of 1.3b, 13b, and 30b, the hidden-size $d$ dimensions are 2048, 5120, and 7168, respectively. Our findings indicate that a rank $r$ of 8 is adequate (as shown in Table 9), leading to added **parameter ratios** of 0.008, 0.003, and 0.002, respectively. For precision clarity, the $W$ matrix is quantized to 4/3/2 bits in our approach, while LoRC components ($U$ and $V$) are maintained at 8 bits. Hence, the memory considerations should indeed reflect this distinction. We provide further details on computational memory and FLOPs as follows:

- **FLOPs Calculation.** For the operation $WX + UVX$, with the input $X$ having dimensions $d \times s$ (where $s$ is the sequence length, typically 512, 1024, or 2048 tokens), the FLOPs for the matrix multiplication $WX$ would be $\mathbf{2sd^2}$. For $UVX$, by first computing $Y = VX$ and then $UY$, the FLOPs required are $2rsd$ for each step. Consequently, the operation $U(VX)$ necessitates a total of $\mathbf{4rsd}$ FLOPs.
- **Memory Impact.** For the $W+UV$ configuration, a single parameter requires either 0.5 byte or 1 byte for 4-bit or 8-bit representation, respectively. Therefore, for a 13b-model with 4-bit precision, the memory requirement is $\mathbf{6.5}$GB. Incorporating an 8-dimensional 8-bit LoRC adds a mere $\mathbf{0.006}$GB to this 6.5GB model.

Significantly, LoRC can be viewed as a supplementary feature to existing quantization methodologies such as RTN, GPTQ, and ZeroQuant-Local/Global, and can be seamlessly integrated with FGQ. We have conducted experiments to evaluate the performance of LoRC on both OPT and BLOOM, applying 4-bit, 3-bit, and 2-bit weights by setting the activation to FP16.[5] Based on the discoveries in the preceding sections, we utilize the GPTQ quantization strategy. To gain

[5]For INT8 Activation, the observation for FP16 holds similarly for INT8 Activation.

| Precision | block-size (W\|A) | OPT-6.7b | OPT-13b | OPT-30b | OPT-66b | BLM-1.7b | BLM-3b | BLM-7.1b | BLM-176b |
|---|---|---|---|---|---|---|---|---|---|
| W4A16 | 128 \| NA | 12.10 | 11.28 | 10.74 | 10.44 | 20.92 | 17.90 | 15.17 | 10.94 |
| W4A8 | Case-1: per-row \| per-row | 13.17 | 13.07 | 14.65 | 16.32 | 21.43 | 18.39 | 15.58 | 11.19 |
| | Case-2: per-row \| 128 | 12.29 | 11.45 | 10.80 | 10.61 | 21.59 | 18.31 | 15.52 | 11.03 |
| | Case-3: 128 \| 128 | 12.04 | 11.31 | 10.75 | 10.45 | 21.27 | 17.86 | 15.19 | 10.96 |

Table 6: OPT W4$^{\text{asym}}$-A8 with block-sizes out of the best result from GPTQ, ZQ-Local, and ZQ-Global on OPT and BLOOM.

| Bits | LoRC | Coarse-grained weight quantization (per-row block-size) | | | | | Fine-grained quantization on weight (256 block-size ) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OPT-6.7b | OPT-13b | OPT-30b | OPT-66b | BLM-176b | OPT-6.7b | OPT-13b | OPT-30b | OPT-66b | BLM-176b |
| W8A16 | | 11.90 | 11.22 | 10.70 | 10.33 | 10.90 | 11.90 | 11.22 | 10.70 | 10.33 | 10.90 |
| W4A16 | ✗ | 12.28 | 11.42 | 10.78 | 10.78 | 11.02 | 12.05 | 11.28 | 10.74 | 10.50 | 10.95 |
| | ✓ | 12.10 | 11.36 | 10.76 | 10.34 | 10.98 | 11.99 | 11.29 | 10.70 | 10.29 | 10.93 |
| W3A16 | ✗ | 14.18 | 12.43 | 11.28 | 17.77 | 49.46 | 12.79 | 11.63 | 10.9 | 11.34 | 11.13 |
| | ✓ | 13.00 | 11.90 | 11.14 | 10.63 | 11.30 | 12.40 | 11.57 | 10.83 | 10.42 | 11.08 |
| W2A16 | ✗ | 120.56 | 40.17 | 25.74 | 225.45 | Explode | 23.13 | 15.55 | 12.68 | 308.49 | 12.64 |
| | ✓ | 24.17 | 18.53 | 14.39 | 13.01 | 14.15 | 16.27 | 14.30 | 12.37 | 11.54 | 12.21 |

Table 7: W#$^{\text{asym}}$-A16 quantization with # being 4-bit, 3-bit and 2-bit on OPT and BLOOM (BLM).

| LoRC | Coarse-grained weight quantization | | | |
|---|---|---|---|---|
| $\hat{U}, \hat{V}$ | 6.7b | 13b | 30b | 66b |
| FP16 | 12.08 | 11.35 | 10.76 | 10.31 |
| INT8 | 12.10 | 11.36 | 10.76 | 10.34 |

Table 8: Results of W4$^{\text{asym}}$ A16 quantization with LoRC approximating $\hat{E} = \hat{U}\hat{V}$ on OPT model family. $\hat{U}$ and $\hat{V}$ can be represented with FP16 or INT8, of which the performance are represented below.

| LoRC-dim $m$ | OPT-1.3b | OPT-6.7b | OPT-30b |
|---|---|---|---|
| $m = 0$ basline | 15.95 | 12.06 | 10.73 |
| $m = 4$ | 15.73 | 12.00 | 10.72 |
| $m = 8$ | 15.76 | 11.99 | 10.70 |
| $m = 16$ | 15.74 | 12.00 | 10.69 |

Table 9: W4A16 quantization with LoRC by varying $m$.

a comprehensive understanding of LoRC, we include the results with and without the application of FGQ. The datasets and hyperparameters are consistent with those detailed in earlier sections.

**Evaluation Results.** The findings are showcased in Table 7, split into two sections: coarse-grained weight quantization (per-row) and fine-grained quantization (block-size 256). Notably, we observe that the two low-rank matrices, $\hat{U}$ and $\hat{V}$, can be quantized to 8-bit without any performance discrepancy (Table 8). Thus, the two low-rank matrices for LoRC in Table 7 are INT8 with $m = 8$.

Several key observations can be made. Firstly, LoRC consistently boosts performance across all bit sizes and block sizes, as indicated by the lower perplexity scores when LoRC is activated. Secondly, the enhancement brought about by LoRC becomes more substantial as the bit size diminishes, especially noticeable for W2A16, which displays a markedly greater impact compared to W4A16 and W3A16 in most scenarios. Lastly, the combination of fine-grained quantization with LoRC yields the most impressive results, underscoring the efficacy of LoRC when integrated with FGQ. Notably, recovering the last 0.05-0.1 perplexity can be challenging, but with LoRC, we are able to nearly recover the original model quality for INT4 quantization.

**Ablation Study on the Low Rank Dimension $m$.** An essential aspect of the LoRC method is on the optimal low-rank dimension, denoted as $m$, explained in **Step II**. To explore this, we varied $m$ in the range of 1, 4, 8, 16, and 32 for OPT-1.3b/6.7b/30b models, and applied W4A16 GPTQ quantization. The outcomes are depicted in Table 9, indicating that the enhancements achieved through LoRC begin to plateau as the dimension $m$ surpasses 4. The most optimal performance for OPT-6.7b is realized when $m = 8$.

## Conclusion

In this work, we provide a comprehensive study of post-training quantization (PTQ) on large language models with different PTQ methods (e.g., RTN, GPTQ, ZeroQuant), and with different quantization coverage (weight-only and weight-and-activation quantization), etc. We find that PTQ methods are critical to improving the quantized model quality, and that fine-grained quantization (FGQ) can bring acceptable accuracy and model size trade-off. Finally, we introduced an optimization technique called Low Rank Compensation (LoRC), which works synergistically with PTQ and FGQ, playing a crucial role in enhancing full model quality recovery with a minimal increase in model size.

## References

Bai, H.; Zhang, W.; Hou, L.; Shang, L.; Jin, J.; Jiang, X.; Liu, Q.; Lyu, M.; and King, I. 2020. BinaryBERT: Pushing the Limit of BERT Quantization. *arXiv preprint arXiv:2012.15701*.

Big-Science. 2022. Bloom Inference. https://github.com/huggingface/transformers-bloom-inference/tree/main/bloom-inference-scripts.

Bondarenko, Y.; Nagel, M.; and Blankevoort, T. 2021. Understanding and overcoming the challenges of efficient transformer quantization. *arXiv preprint arXiv:2109.12948*.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Copilot. 2021. GitHub Copilot. https://github.com/features/copilot/.

Darvish Rouhani, B.; Lo, D.; Zhao, R.; Liu, M.; Fowers, J.; Ovtcharov, K.; Vinogradsky, A.; Massengill, S.; Yang, L.; Bittner, R.; et al. 2020. Pushing the limits of narrow precision inferencing at cloud scale with microsoft floating point. *Advances in neural information processing systems*, 33: 10271–10281.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Dettmers, T.; and Zettlemoyer, L. 2022. The case for 4-bit precision: k-bit Inference Scaling Laws. *arXiv preprint arXiv:2212.09720*.

Esser, S. K.; McKinstry, J. L.; Bablani, D.; Appuswamy, R.; and Modha, D. S. 2019. Learned step size quantization. *arXiv preprint arXiv:1902.08153*.

Fan, A.; Stock, P.; Graham, B.; Grave, E.; Gribonval, R.; Jegou, H.; and Joulin, A. 2020. Training with quantization noise for extreme fixed-point compression. *arXiv preprint arXiv:2004.07320*.

Frantar, E.; and Alistarh, D. 2022. Optimal Brain Compression: A framework for accurate post-training quantization and pruning. *arXiv preprint arXiv:2208.11580*.

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*.

Gholami, A.; Yao, Z.; Kim, S.; Mahoney, M. W.; and Keutzer, K. 2021. AI and Memory Wall. *RiseLab Medium Post*.

Hassibi, B.; and Stork, D. G. 1993. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in neural information processing systems*, 164–171.

Hinton, G.; Vinyals, O.; and Dean, J. 2014. Distilling the knowledge in a neural network. *Workshop paper in NIPS*.

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Kim, S.; Gholami, A.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2021. I-bert: Integer-only bert quantization. In *International conference on machine learning*, 5506–5518. PMLR.

LeCun, Y.; Denker, J. S.; and Solla, S. A. 1990. Optimal brain damage. In *Advances in neural information processing systems*, 598–605.

Marcinkiewicz, M. A. 1994. Building a large annotated corpus of English: The Penn Treebank. *Using Large Corpora*, 273.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

OpenAI. 2022. OpenAI Chatgpt. https://openai.com/blog/chatgpt/.

Polino, A.; Pascanu, R.; and Alistarh, D. 2018. Model compression via distillation and quantization. *arXiv preprint arXiv:1802.05668*.

Pope, R.; Douglas, S.; Chowdhery, A.; Devlin, J.; Bradbury, J.; Levskaya, A.; Heek, J.; Xiao, K.; Agrawal, S.; and Dean, J. 2022. Efficiently Scaling Transformer Inference. *arXiv preprint arXiv:2211.05102*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683.

Scao, T. L.; Fan, A.; Akiki, C.; Pavlick, E.; Ilić, S.; Hesslow, D.; Castagné, R.; Luccioni, A. S.; Yvon, F.; Gallé, M.; et al. 2022. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. *arXiv preprint arXiv:2211.05100*.

Shen, S.; Dong, Z.; Ye, J.; Ma, L.; Yao, Z.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2020. Q-BERT: Hessian Based Ultra Low Precision Quantization of BERT. In *AAAI*, 8815–8821.

Smith, S.; Patwary, M.; Norick, B.; LeGresley, P.; Rajbhandari, S.; Casper, J.; Liu, Z.; Prabhumoye, S.; Zerveas, G.; Korthikanti, V.; et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Tao, C.; Hou, L.; Zhang, W.; Shang, L.; Jiang, X.; Liu, Q.; Luo, P.; and Wong, N. 2022. Compression of Generative Pre-trained Language Models via Quantization. *arXiv preprint arXiv:2203.10705*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wu, X.; Li, C.; Aminabadi, R. Y.; Yao, Z.; and He, Y. 2023. Understanding INT4 Quantization for Transformer Models: Latency Speedup, Composability, and Failure Cases. *arXiv preprint arXiv:2301.12017*.

Wu, X.; Yao, Z.; Zhang, M.; Li, C.; and He, Y. 2022. Extreme compression for pre-trained transformers made simple and efficient. *arXiv preprint arXiv:2206.01859*.

Xiao, G.; Lin, J.; Seznec, M.; Demouth, J.; and Han, S. 2022. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. *arXiv preprint arXiv:2211.10438*.

Yao, Z.; Aminabadi, R. Y.; Zhang, M.; Wu, X.; Li, C.; and He, Y. 2022. ZeroQuant: Efficient and Affordable Post-Training

Quantization for Large-Scale Transformers. *arXiv preprint arXiv:2206.01861*.

Zadeh, A. H.; Edo, I.; Awad, O. M.; and Moshovos, A. 2020. Gobo: Quantizing attention-based nlp models for low latency and energy efficient inference. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 811–824. IEEE.

Zafrir, O.; Boudoukh, G.; Izsak, P.; and Wasserblat, M. 2019. Q8BERT: Quantized 8bit bert. *arXiv preprint arXiv:1910.06188*.

Zeng, A.; Liu, X.; Du, Z.; Wang, Z.; Lai, H.; Ding, M.; Yang, Z.; Xu, Y.; Zheng, W.; Xia, X.; et al. 2022. GLM-130B: An Open Bilingual Pre-trained Model. *arXiv preprint arXiv:2210.02414*.

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*.

Zhang, W.; Hou, L.; Yin, Y.; Shang, L.; Chen, X.; Jiang, X.; and Liu, Q. 2020. Ternarybert: Distillation-aware ultra-low bit bert. *arXiv preprint arXiv:2009.12812*.