LLMC: Benchmarking Large Language Model Quantization with a Versatile Compression Toolkit

Ruihao Gong [1,2*] **Yang Yong** [2*] **Shiqiao Gu** [2*] **Yushi Huang** [1,2*]
Chengtao Lv[1,2] **Yunchen Zhang**[2] **Dacheng Tao**[3] **Xianglong Liu** [1†]
[1]Beihang University [2]SenseTime Research [3]Nanyang Technological University
{gongruihao, yongyang, gushiqiao, huangyushi, lvchengtao, zhangyunchen}@sensetime.com
xlliu@buaa.edu.cn dacheng.tao@ntu.edu.sg

Abstract

Recent advancements in large language models (LLMs) are propelling us toward artificial general intelligence with their remarkable emergent abilities and reasoning capabilities. However, the substantial computational and memory requirements limit the widespread adoption. Quantization, a key compression technique, can effectively mitigate these demands by compressing and accelerating LLMs, albeit with potential risks to accuracy. Numerous studies have aimed to minimize the accuracy loss associated with quantization. However, their quantization configurations vary from each other and cannot be fairly compared. In this paper, we present LLMC, a plug-and-play compression toolkit, to fairly and systematically explore the impact of quantization. LLMC integrates dozens of algorithms, models, and hardware, offering high extensibility from integer to floating-point quantization, from LLM to vision-language (VLM) model, from fixed-bit to mixed precision, and from quantization to sparsification. Powered by this versatile toolkit, our benchmark covers three key aspects: calibration data, algorithms (three strategies), and data formats, providing novel insights and detailed analyses for further research and practical guidance for users. Our toolkit is available at `https://github.com/ModelTC/llmc`.

1 Introduction

Recently, LLMs such as GPT-4 (OpenAI et al., 2024) have demonstrated unprecedented generative capabilities in the field of natural language processing (NLP) and also achieved widespread applications. However, their substantial computational and storage costs have impeded their further popularization among users. For instance, BLOOM (Touvron et al., 2023), a multilingual LLM with 176 billion parameters, requires a minimum of 350 GB space to store model weights in full-precision (FP16) format. Even worse, it requires 5×80GB A100 or 9×40GB A800 NVIDIA GPUs to perform inference. Therefore, reducing LLMs' serving cost is paramount to further enhance their application.

For the aforementioned challenge, model quantization (Nagel et al., 2021) can be an effective solution. It maps weights and/or activations to a lower-bit data format to reduce memory footprint and accelerate model inference. Existing quantization approaches can be categorized into two types: quantization-aware-training (QAT) (Bhalgat et al., 2020; Gong et al., 2019; Esser et al., 2020) and post-training quantization (PTQ) (Wei et al., 2023a; Li et al., 2021). Although with prominent high performance, the necessity for QAT to undergo finetuning or retraining with substantial training data and training costs renders it unattainable for the majority of users. Correspondingly, PTQ compresses models without retraining, making it a preferred method for LLMs due to its minimal resource requirements. Therefore, we do not mention some QAT methods (Du et al., 2024; Liu et al., 2024, 2023b; Egiazarian et al., 2024) in this paper.

However, current PTQ methods always evaluate across distinct datasets in different quantization configurations and with simulated quantization. For example, AWQ (Lin et al., 2023) employs Pile (val) (Gao et al., 2020a) as calibration data, instead of C4 (Raffel et al., 2019) in GPTQ (Frantar et al., 2022). This situation would cause an inaccurate assessment of configurations for efficient and accurate LLM quantization.

To provide a comprehensive options menu for users and directions with insights for further research, we make a fair benchmark, which considers three key dimensions, *e.g.*, calibration data, algorithms, and data formats. First, we systematically explore the effect of calibration data for higher model performance. Then, we aim to investigate the effectiveness and underlying mechanisms of

*Equal contribution.
†Corresponding authors.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 132–152
November 12-16, 2024

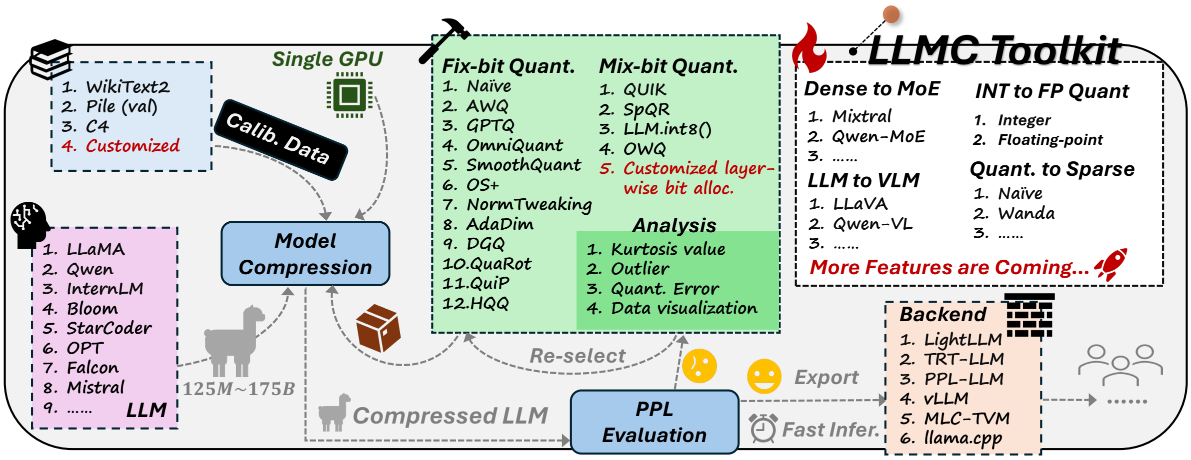

Figure 1: Overview of our LLM compression toolkit LLMC, which incorporates diverse algorithms, ultra-low cost quantization, multiple backends support, and high extensibility. More features are under development.

three primary algorithm strategies: transformation, clipping, and reconstruction. Finally, we probe how to select types between the integer and float-point quantization for further accuracy improvements. All the aforementioned studies benefit from our LLMC, a user-friendly, plug-and-play LLM compression toolkit. This toolkit incorporates several distinct traits, as demonstrated in Figure 1, offering users the freedom to select options that best suit their needs.

In a word, our main contributions can be described as follows:

- We release a versatile LLM compression toolkit LLMC supporting dozens of algorithms, models, and multiple inference backends with powerful expandability and all-around evaluation. It also enables users to perform compression for 100-billion-parameter LLMs with just a single GPU, which substantially facilitates the application of LLM quantization.
- We modularly and fairly benchmark LLM quantization considering calibration data, algorithms, and data formats. With detailed observation and analysis, we provide various types of novel points for performance and method improvements under different configurations.
- Equipped with our powerful toolkit and comprehensive insights, future LLM researchers can efficiently integrate suitable algorithms and low-bit formats for their applications, thereby democratizing the compression of large language models.

2 LLMC: A Versatile LLM Compression Toolkit

First and foremost, we have developed a comprehensive toolkit named LLMC for LLM compression, characterized by the following key features, which are also exhibited in Figure 1.

Diverse algorithms support. LLMC supports a wide range of quantization algorithms, including 16 different methods covering weight-only, weight-activation, and mixed-precision quantization. This variety allows for fair comparisons and in-depth analyses of different approaches.

Quantization with an ultra-low cost. Our toolkit is designed to be resource-efficient, and capable of running large models with minimal hardware requirements. Benefiting from our pipeline with offloading technique, only one 40G A100 is required to calibrate and evaluate OPT-175B (Zhang et al., 2022), whose weights occupies ≈ 350GB.

Multi-backend compatibility. Built on LLMC, various quantization settings and model formats are compatible with multiple backends and hardware platforms, such as LightLLM (ModelTC, 2023), TRT-LLM (Nvidia, 2023), PPL-LLM (OpenPPL, 2023), vLLM (Kwon et al., 2023), MLC-LLM (team, 2023), and llama.cpp (llama.cpp team, 2023), making it highly versatile.

High extensibility. The toolkit is highly modular and extensible, allowing easy adaptation [1] from integer quantization to floating-point quantization, from LLMs to VLMs (Zhang et al., 2024), from quantization to sparsification, and from dense models to Mixture-of-Expert (MoE) models (Shazeer et al., 2017). This modularity ensures users can extend and customize the toolkit to meet their needs.

Comprehensive evaluation. LLMC enables comprehensive evaluation of quantized models, providing detailed performance metrics and analysis, *e.g.*, PPL (Alon and Kamfonas, 2023), and data

[1]All adaptations mentioned here have been implemented and results are shown in the appendix.

visualization analysis, *e.g.,* Kurtosis value, quantization error, and outlier distribution. This thorough evaluation capability ensures that users can make informed decisions about the best quantization strategies for their models.

3 Benchmarking LLM Quantization

Powered by LLMC toolkit, we explore the quantization of LLMs from three distinct perspectives: the calibration data in subsection 3.2, the algorithms in subsection 3.3, and the data format of quantization in subsection 3.4. More explorations, *e.g.*, extendability of LLMC, KV cache quantization, and inference speed can be found in the appendix.

3.1 Experimental Settings

We first introduce experimental settings as follows. More implemental details with quantization preliminary can be found in the appendix.

Models. To demonstrate the generability of our benchmark, we access performance on LLaMA-2 (Touvron et al., 2023) and LLaMA-3 (AI@Meta, 2024) family, spanning model sizes from 7B to 70B for general language tasks. To broaden the scope of our evaluation, we show more results in the appendix, including ChatGLM (Zeng et al., 2023) for long context abilities, LLaVA-1.5 (Liu et al., 2023a) for the multimodal task, Mixtral (Jiang et al., 2024) as a representative of MoE models.

Datasets. We categorize the evaluation datasets into upstream and downstream datasets. For the upstream datasets, we employ WikiText2 (Foundation) and C4 (Raffel et al., 2019) dataset with the perplexity metric for evaluation, since perplexity can stably reflect the LLM's perfomance (Dettmers and Zettlemoyer, 2023). For the downstream tasks, we select examination tasks including MMLU (Hendrycks et al., 2021), ARC-e (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), GPQA (Rein et al., 2023), MBPP (Austin et al., 2021), Human-Eval (Chen et al., 2021a), the long context evaluation LongBench (Bai et al., 2023), and multimodal evaluation MME (Fu et al., 2023). For the calibration data, to ensure a fair comparison, the vast majority of experiments use the same subset of the Pile (Gao et al., 2020b) validation set. We use the same calibration data number of 128 and the same sequence length of 512. We also find that different preprocessing methods of the calibration data can affect the quantization accuracy significantly. So, we use the same preprocessing method as in our open-source code.

3.2 Impact of Calibration Data

With fair experimental settings, we first explore how calibration data impacts quantization accuracy. Prior studies (Li et al., 2023; Liu et al., 2023b) highlight significant effects of different calibration datasets on quantized model performance. Yet, a systematic analysis of crucial factors is lacking. To address this, we identify and propose two key aspects to guide future calibration data selection.

Token distribution consistency. Previous research (Cai et al., 2020; Zhang et al., 2021) focuses on synthesizing better distribution-matched calibration images to achieve higher performance for vision models. Derived from that view, we are the first to investigate the impact of the token distribution relationship between calibration and test data on model performance. As shown in Table 1 and Figure 2, we find that the performance of a model calibrated with data that more closely matches the token distribution of the test set tends to be superior. For instance, WikiText2 calibration data with 1.97 lower D_{KL} achieves a ≈ 0.2 PPL decrease than Pile (val) on the WikiText2 test data with GPTQ quantization. This finding indicates the importance of selecting calibration data with an aligned distribution for the data in practice.

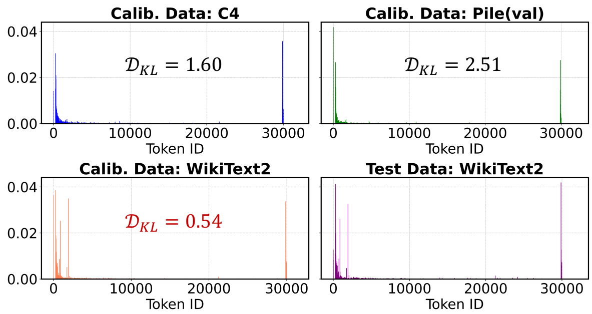

Figure 2: Token distribution for calibration/test datasets. The y-axis shows frequency, the x-axis shows token ID, and "$\mathcal{D}_{KL}$" calculates the KL divergence between the calibration data and the specific test data: WikiText2.

Calib. Data	GPTQ	AWQ	OmniQuant
C4	6.323	6.173	5.717
Pile (val)	6.330	6.195	5.753
WikiText2	**6.133**$_{+0.568}$	**6.144**$_{+0.156}$	**5.697**$_{+0.516}$

Table 1: Impact of calibration data on performance across algorithms. We evaluate the PPL↓ of WikiText2 test data, employing w3a16g128 GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2023), and w6a6 OmniQuant (Shao et al., 2023) quantized LLaMA-2-7B. Data indices show differences in results from randomly shuffling token order within each data entry.

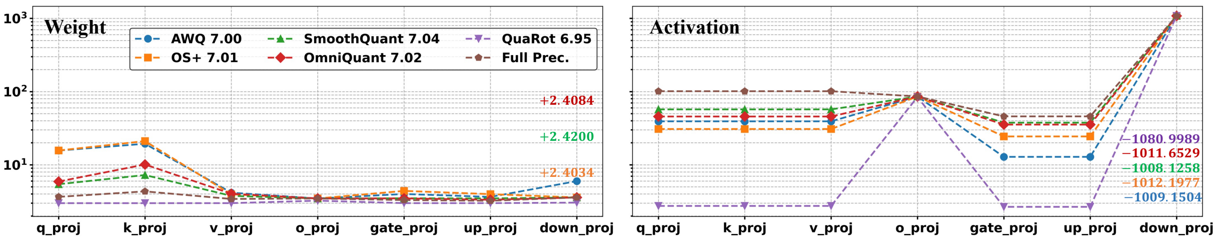

Figure 3: Kurtosis value of weights (Left) and input activations (Right) with various layer types for different methods under w6a6 quantization. The legends denote the quantization method and its corresponding PPL on WikiText2. We do not employ transformation for `down_proj` for a fair comparison, as only default AWQ and QuaRot include this position. The colorful values represent changes of K after using transformation for `down_proj` for all scaling-based methods, and online transformation for QuaRot. To be noted, we only mark numbers > 0.2 for all the cases.

Intra-sentence logic. Unlike vision models that utilize image calibration data, LLMs' calibration data consist of sequentially ordered token sequences that embody logical meaning. Therefore, we also conduct experiments to explore the impact of that logic on LLM quantization. Seeing from the data indices in Table 1, breaking the logic within the calibration data can cause a non-negligible accuracy drop. Notably, in this scenario, the robustness of learning/reconstruction-based algorithms such as GPTQ, and OmniQuant are lower than non-learning methods. Specifically, both exhibit $\times 3.3$ PPL increasing compared with AWQ. Overall, people should not seek or generate an illogical corpus to calibrate LLMs.

3.3 Dive into the Quantization Algorithms

Besides calibration data, we could also methodically explore and benchmark LLM quantization algorithms equipped with our LLMC. Three main techniques for the field are outlier transformation, weight clipping, and weight reconstruction. However, how and how much they help under different scenarios remains unclear, as existing studies lack fair comparisons. Therefore, we will respectively discuss these methods in this section.

3.3.1 How Does Transformation Influence Activation and Weight Outlier?

Most of the existing works aim to reduce the outliers via different kinds of equivalent transformation [2], which can be categorized as scaling-based transformation, *e.g.*, AWQ (Lin et al., 2023), SmoothQuant (Xiao et al., 2023), OS+ (Wei et al., 2023b), and OmniQuant (Shao et al., 2023) and rotation-based transformation, for instance, QuaRot (Ashkboos et al., 2024).

Scaling-based transformation typically involves searching for or learning a scaling vector to convert activation outliers into weights by optimizing the layer's quantization error. Conversely, the rotation-based transformation employs an Orthogonal matrix without accounting for output error. To thoroughly examine their effects, we analyze the kurtosis value [3] of each layer after transformation, providing insights into their inherent mechanisms.

From Figure 3 and Table 2, We observe three distinct findings. *1)* Scaling-based transformation methods achieve lower K for activations at the cost of higher K for weights compared with full precision, which would induce a non-negligible performance degradation for lower-bit weight quantization, even with higher-bit activations can not eliminate the risk (w6a6 > w4a8 in Table 3). *2)* K for some specific positions like `down_proj` layers is significantly higher than others. These positions have a pronounced impact on accuracy. For example, with `down_proj` transformed (evident lower K in Figure 3), salient improvements are gained as exhibited in Table 3. *3)* Although the rotation-based transformation reduces outliers by directly optimizing the tensor's outliers, it may not realize obvious accuracy improvement in some cases. From Table 2, it is evident that the quantization error of output tensors is not minimized, as optimization did not focus on reducing output error, leading to a higher PPL.

3.3.2 When Should We Utilize the Weight Clipping?

The technique of weight clipping, restricting the range of weight values before quantization, has been recognized for its contribution to maintaining better performance (Lin et al., 2023; Du et al., 2024; Shao et al., 2023) for the quantization process. Here, we analyze its application situations

[2]In this section, our experiments only employ transformation methods in each algorithm. We also apply transformation of AWQ to weight activation quantization.

[3]Kurtosis value is defined as $K = \frac{1}{n}\sum_{i=1}^{n}(\frac{\boldsymbol{X}_i-\mu}{\sigma})^4$, where μ and σ represent mean and variance of a tensor $\boldsymbol{X}$, to reflect outlier conditions (Bondarenko et al., 2023).

Method	q_proj	k_proj	v_proj	o_proj	gate_proj	up_proj	down_proj	PPL↓
Full Prec.	3.6505	4.3354	3.4174	3.4720	3.2991	3.2300	3.5845	6.14
AWQ	4.9219	6.1633	3.4602	3.4720	3.3190	3.2438	4.3083	8.57
	0.9960	0.9960	0.9784	0.9387	0.9882	0.9628	0.9479	
QuaRot	2.9051	2.9050	2.9069	2.9075	2.9074	2.9073	2.9075	40.81
	0.9962	0.9967	0.9797	0.8286	0.9764	0.9579	0.9230	

Table 2: Comparison on K and PPL on Wikitext2 of w3a16g128 LLaMA-3-8B for scaling-based transformation methods AWQ and rotation-based transformation method QuaRot. Due to the neglect of optimizing output quantization error (cosine similarity in the gray cells), QuoRot results in higher PPL even with fewer outlier issues.

AWQ		SmoothQuant		OS+		OmniQuant		QuaRot	
w4a8	w6a6	w4a8	w6a6	w4a8	w6a6	w4a8	w6a6	w4a8	w6a6
8.60	7.00	8.85	7.04	8.55	7.01	8.83	7.02	9.77	6.95
7.77	6.79	7.92	6.85	7.76	6.81	7.92	6.83	9.43	6.74

Table 3: PPL on Wikitext2 for different transformation methods with or without transforming down_proj layers for LLaMA-3-8B. The gray raw indicates the results are obtained with down_proj layers transformed.

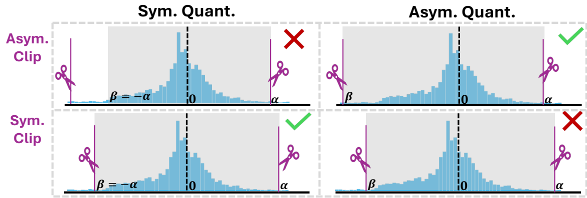

Figure 4: Comparison between asymmetric and symmetric weight clipping *w.r.t.* asymmetric/symmetric quantization. After weight clipping, we obtain the final range of tensor to quantize as depicted in the solid gray box related to asymmetric/symmetric quantization.

#Bits	Method	LLaMA-2-7B		LLaMA-2-70B	
		Avg. PPL↓	Avg. Acc.↑	Avg. PPL↓	Avg. Acc.↑
w3a16g128	AWQ	7.25	61.18	4.90	80.95
	AWQ *w/* asym. clip	7.21	61.59	4.89	81.07
w2a16g64	AWQ	1.8e5	37.69	6.8e4	32.84
	AWQ *w/* asym. clip	13.26	48.77	6.49	75.31

Table 4: Impact of asymmetric/symmetric weight clipping. We evaluate the average accuracy and the average PPL here. "asym. clip" means we employ asymmetric clipping.

under two different scenarios.

Symmetric or asymmetric. Clipping and quantization can be divided into symmetric or asymmetric categories. However, previous studies (Lin et al., 2023; Liu et al., 2024) always neglect their relationships and employ wrong patterns. As shown in Figure 4, we can observe that symmetric clipping with symmetric quantization maintains more information (*i.e.*, solid gray box) than with asymmetric quantization, and for asymmetric clipping vice versa. This finding can help improve current methods with significant accuracy recovery, especially for extremely lower bit-width. For instance, in Table 4, default AWQ, applying asymmetric quantization with symmetric clipping, results in a 6.8e4 PPL score and performance [4] declines of 48.11% for 2-bit LLaMA-2-70B compared with 3-bit configuration. Conversely, equipping with asymmetric clipping, AWQ in LLMC achieves 42.47% accuracy upswings with admissible PPL.

Bit-width. Besides different combinations of quantization and clipping, we also investigate the impact of clipping with different bit-width. From Table 5, weight clipping does not show superiority across all bit-widths. *1) For higher bit (4-bit) weight-only quantization, clipping has a side-effect, unlike improvement for lower-bit (3-bit).* We hypothesize that in 4-bit quantization, weight clipping causes more information loss than quantization rounding. However, for 3-bit quantization, quantization rounding has a greater impact. *2) For weight activation quantization, suitable clipping exhibits positive effects whatever bit-width.* We ascribe this for clipping anomalous values effectively adjusting the majority of weights (*i.e.*, moderate and small elements). Accounting for hard-quantized and considerably influential activations, this approach significantly reduces the output errors resulting from the multiplication of quantized large activations with well-adjusted weights [5], which greatly reduce the impact of these quantized activations.

3.3.3 Should We Combine Transformation and Reconstruction?

Apart from transformation and clipping, the reconstruction-based method like GPTQ (Frantar et al., 2022) is also widely used to quantize weights. This method iteratively updates the unquantized weights to compensate for the impact of the current quantized weights, thereby minimizing the output quantization error. Some recent transformation methods (Ashkboos et al., 2024; Lin et al., 2023) integrate this technique to demonstrate their extendability.

Nevertheless, we find that a significant and obvious accuracy from this combination is not usually the case. From Table 6 [6], we conclude that: *1)* the

[4] Without special claims, we calculate average accuracy on five downstream tasks: MMLU, ARC-e, BoolQ, HellaSwag, and PIQA, and average PPL on WikiText2 and C4 in the paper. Detailed data is presented in the appendix subsection A.7.

[5] Activation outliers make huge performance deterioration can be found in `LLM.int8()` (Dettmers et al., 2022).

[6] Clipping for AWQ here is canceled to expel distractions.

Model	w3a16g128		w4a16g128		w6a6		w8a8	
	w/ clip	*w/o* clip	*w/* clip	*w/o* clip	*w/* clip	*w/o* clip	*w/* clip	*w/o* clip
LLaMA-3-8B	11.74	11.23	11.99	17.42	10.35	9.46	10.73	10.35
	30.60	24.80	40.60	42.20	40.60	39.40	43.80	43.80
LLaMA-3-70B	8.08	7.57	9.09	11.62	26.38	25.75	16.79	16.66
	54.00	54.20	59.20	60.00	58.20	58.20	60.20	57.60

Table 5: Impact of weight clipping under various bit-width. We employ AWQ for weight-only and OS+ for weight activation quantization with or without clipping as methods here. Accuracy on GPQA is highlighted in gray rows, and the rest for MBPP.

Metric	GPTQ	AWQ	AWQ *w/* GPTQ	QuaRot	QuaRot *w/* GPTQ
Avg. PPL↓	10.67	10.98	10.55	50.00	10.35
Avg. Acc.↑	71.96	70.72	72.72	45.90	74.84

Table 6: Impact of reconstruction (GPTQ) combined with scaling (AWQ) and rotation-based (QuaRot) transformations for w3a16g128 LLaMA-3-8B.

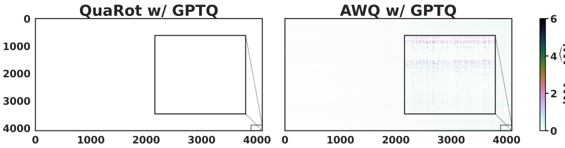

Figure 5: Visualization of relative quantization errors for the weight of `q_proj` in the first block for w3a16g128 LLaMA-3-8B. $\widehat{\boldsymbol{W}}$ represents the quantized counterpart of the weight $\boldsymbol{W}$.

scaling-based transformation like AWQ *w/* GPTQ shows moderate improvement for LLaMA-3-8B. *2)* However, The rotation-based method QuaRot *w/* GPTQ far surpasses QuaRot alone, even with 28.94% accuracy boost for 3-bit LLaMA-3-8B. The inherent reason might lie in two aspects: *1)* Scaling-based transformation methods may amplify weight outliers [7]. This gives rise to a larger challenge for iterative compensation during the reconstruction, especially weights in rear columns which GPTQ can not properly deal with [8]. However, QuaRot, which effectively eliminates weight outliers, pairs well with GPTQ. From Figure 5, the steeper quantization error of later weight columns for AWQ *w/* GPTQ compared with QuaRot *w/* GPTQ validates our analysis. *2)* Rotation-based transformation only aims to decrease tensor outliers without considering output errors, so the kurtosis value is significantly reduced. However, for weight-only quantization, outliers in the activation might amplify the error in quantized weights [9], leading to obvious output discrepancy. GPTQ exactly considers the output error through approximated Hessian matrix, and thus can always complement rotation-based transformation. As in Table 7, QuaRot *w/* GPTQ performing a much higher cosine similarity between the output of the corresponding layer and its quantized counterpart helps confirm our analysis.

[7] K analysis in subsubsection 3.3.1 verifies this.
[8] This can be found in QUIK (Ashkboos et al., 2023)
[9] The importance of salient activation is described in AWQ.

Method	`q_proj`	`k_proj`	`v_proj`	`o_proj`	`gate_proj`	`up_proj`	`down_proj`
QuaRot	0.9962	0.9967	0.9797	0.8286	0.9764	0.9579	0.9230
QuaRot *w/* GPTQ	0.9971	0.9975	0.9847	0.9476	0.9895	0.9791	0.9529

Table 7: Fine-grained analysis comparing QuaRot and QuaRot *w/* GPTQ in w3a16g128 LLaMA-3-8B. We report the output cosine similarity between the original layer and the quantized layer.

Full Prec.	w3a16g128		w4a16g128		w4a16		w4a4		w6a6	
	Naive	AWQ	Naive	AWQ	Naive	AWQ	Naive	SmoothQuant	Naive	SmoothQuant
5.47	6.66	6.19	5.78	5.59	6.11	5.81	NaN	NaN	6.86	6.77
	6.89	6.38	5.70	5.63	5.89	5.75	90.85	16.35	5.56	5.56

Table 8: PPL for LLaMA-2-7B weight-only quantization and weight-activation INT (gray rows)/FP (whight rows) quantization on WikiText2. Naive means simple round-to-nearest quantization.

3.4 Integer or Floating-point Quantization?

The above-mentioned algorithms are based on integer (INT) quantization. Although traditional INT quantization has received widespread adoption in the industry, floating-point (FP) quantization has emerged as a rising alternative. This is attributed to its superior accuracy and high flexibility, offering advantages for handling long-tailed distributions.

Table 8 reports the detailed FP quantization results for LLMs. For the weight-activation quantization, FP quantization consistently surpasses INT quantization by a large margin as it can better overcome the outlier issue. It is worth noting that under w4a4, the INT quantization suffers from non-trivial performance degradation while FP quantization improves to a usable level. Conversely, when applying weight-only quantization, the FP quantization achieves worse performance under ultra-low-bit ($\leq$ 3-bit) or small group size. These findings indicate that: *1)* the positive zero and negative zero in FP format constrain the representation capability of this quantization type, particularly under low-bit. *2)* the range of small group size is more uniform, which is unsuitable for FP quantization. *3)* the symmetric FP quantization struggles to deal with the asymmetry in LLMs.

4 Additional Results and Discussions

Impact of quantization for fine-tuning. We conduct experiments for quantization on LLaMA-3-8B with supervised fine-tuning (SFT) on Evol-instruction-66k [10] to analyze the impact. We choose ms-swift (Zhao et al., 2024) as the finetuning framework. Additionally, we set the learning rate to 2e-6 with a mini-batch size of 2 and trained the model for 1 epoch on 16 40G A800 GPUs. After fine-tuning, we employ w4a16 naive quantization and AWQ to quantize the model. We choose HumanEval (Chen et al., 2021b) and HumanEval-X (Zheng et al., 2023) for evaluation. As illustrated in Table 9, quantization leads to more severe accuracy drops for the SFT model than the base model. This might be caused by the limited fine-tuning data and more in-depth analyses are needed in the future. Moreover, an advanced algorithm, *i.e.*, AWQ brings obvious improvements compared to Naive quantization for the SFT model.

Test Data	Base/SFT	Base/SFT+Naive	Base/SFT+AWQ
HumanEval	23.78/49.39	19.51/42.07	21.34/46.34
HumanEval-X	32.81/41.58	26.47/36.10	26.83/39.27

Table 9: Accuracy of Base/SFT models after quantization. "Base" denotes LLaMA-3-8B. We report the average accuracy of 5 languages in HumanEval-X.

Impact of calibration data for VLMs. Besides LLMs, we further present the impact of calibration data for LLaVA-7B (Liu et al., 2023a) here. The results in Table 10 indicate that we should collect text and vision data together for VLM quantization.

Method	Perception	Cognition
FP	1477.60	283.21
Calib. Data: Pile (val)	1437.94	274.64
Calib. Data: T&V	1470.93	286.78

Table 10: Impact of calibration data for VLMs. We employ w4a16 AWQ. "T&V" denotes MS-COCO (Lin et al., 2014) and TextVQA (Singh et al., 2019).

Accuracy alignment with the existing methods. Except for the PPL alignment results in subsection A.3, we further conduct downstream experiments for LLaMA-2-7B to prove our reproducibility (experimental details in the appendix). As illustrated in Table 11 and Table 12, our LLMC is reliable in reproducing the outcomes of existing quantization methods.

w4a16g128	MMLU	BoolQ	ARC-e	PIQA
AWQ	46.36	71.25	54.14	77.04
AWQ-LLMC	46.47	71.62	53.96	77.26
GPTQ	43.36	72.81	51.50	77.86
GPTQ-LLMC	43.40	72.91	51.50	77.75

Table 11: Alignment for weight-only quantization. "-LLMC" represents the results are reproduced with our toolkit LLMC.

w8a8	MMLU	BoolQ	ARC-e	PIQA
SmoothQuant	46.17	69.76	49.03	77.26
SmoothQuant-LLMC	46.28	69.08	50.97	77.26
QuaRot *w/* GPTQ	46.38	71.50	52.73	77.75
QuaRot-LLMC + *w/* GPTQ-LLMC.	46.42	70.61	53.26	77.97

Table 12: Alignment for weight-activation quantization.

Role of model scales. Besides LLaMA-2 and LLaMA-3 families, we also conduct experiments for quantizing different LLM families, *e.g.*, SmolLM-135M/350M/1.7B [11], MiniCPM-1B/2B (Hu et al., 2024), and Qwen-2-0.5B/1.5B (Yang et al., 2024) in subsection A.8. We find that low-bit quantization causes more performance degradation for homology models with a larger size. This phenomenon is counter-intuitive and needs to be further explored. Besides, higher precision quantization, *e.g.*, w8a8 or w4a16 leads to subtle accuracy drops for LLMs across all sizes. We will explore the role of scale for larger LLMs in the future.

Pipeline of LLMC. Basically, our LLMC receives an FP LLM and calculates its quantization parameters with advanced algorithms. Finally, this tool can export the model with quantization parameters to the quantization format compatible with a specific backend like vLLM (Kwon et al., 2023). The detailed usage can be found in the official document [12]. Additionally, LLMC can provide quantization analyses and PPL evaluations for those quantized LLMs. With this tool, people can produce various compressed industrial models deployed on different hardware [13].

5 Conclusion

This paper introduces LLMC, a user-friendly and versatile toolkit for LLM compression. Supported by the toolkit, a series of observations and analyses were conducted, providing valuable and novel insights and suggestions for the community.

Acknowledgements

We sincerely thank the anonymous reviewers for their serious reviews and valuable suggestions. This work was supported by the Beijing Municipal Science and Technology Project (No. Z231100010323002).

[10] https://huggingface.co/datasets/codefuse-ai/Evol-instruction-66k

[11] https://huggingface.co/blog/smollm

[12] https://llmc-en.readthedocs.io/en/latest/

[13] Inference efficiency of compressed models can be found in subsection A.6.

References

AI@Meta. 2024. Llama 3 model card.

Gabriel Alon and Michael Kamfonas. 2023. Detecting language model attacks with perplexity. *Preprint*, arXiv:2308.14132.

Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. 2023. Quik: Towards end-to-end 4-bit inference on generative large language models. *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. *Preprint*, arXiv:2404.00456.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. 2020. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. *Preprint*, arXiv:2004.09576.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2023. Quantizable transformers: Removing outliers by helping attention heads do nothing. *Preprint*, arXiv:2306.12929.

Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. 2020. Zeroq: A novel zero shot quantization framework. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13166–13175.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021a. Evaluating large language models trained on code.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021b. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

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. *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:1803.05457v1*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. `https://github.com/open-compass/opencompass`.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Tim Dettmers and Luke Zettlemoyer. 2023. The case for 4-bit precision: k-bit inference scaling laws. *Preprint*, arXiv:2212.09720.

Dayou Du, Yijia Zhang, Shijie Cao, Jiaqi Guo, Ting Cao, Xiaowen Chu, and Ningyi Xu. 2024. Bitdistiller: Unleashing the potential of sub-4-bit llms via self-distillation. *Preprint*, arXiv:2402.10631.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. 2024. Extreme compression of large language models via additive quantization. *Preprint*, arXiv:2401.06118.

Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. 2020. Learned step size quantization. *Preprint*, arXiv:1902.08153.

Wikimedia Foundation. Wikimedia downloads.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, and Rongrong Ji. 2023. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020a. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020b. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. 2019. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *The IEEE International Conference on Computer Vision (ICCV)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. 2024. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Liang Li, Qingyuan Li, Bo Zhang, and Xiangxiang Chu. 2023. Norm tweaking: High-performance low-bit quantization of large language models. *arXiv preprint arXiv:2309.02784*.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. 2021. Brecq: Pushing the limit of post-training quantization by block reconstruction. *Preprint*, arXiv:2102.05426.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. 2023. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

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

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. 2024. Qllm: Accurate and efficient low-bitwidth quantization for large language models. *Preprint*, arXiv:2310.08041.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023b. Llm-qat: Data-free quantization aware training for large language models. *Preprint*, arXiv:2305.17888.

llama.cpp team. 2023. llama.cpp.

Man Luo, Shuguang Chen, and Chitta Baral. 2021. A simple approach to jointly rank passages and select relevant sentences in the obqa context. *arXiv preprint arXiv:2109.10497*.

ModelTC. 2023. Lightllm. `https://github.com/ModelTC/lightllm`.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. 2021. A white paper on neural network quantization. *Preprint*, arXiv:2106.08295.

Nvidia. 2023. Tensorrt-llm. `https://github.com/NVIDIA/TensorRT-LLM`.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

OpenPPL. 2023. Ppl-llm. `https://github.com/openppl-public/ppl.nn.llm`.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *Preprint*, arXiv:2311.12022.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2023. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *Preprint*, arXiv:1701.06538.

Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8815–8821.

Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8317–8326.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2024. A simple and effective pruning approach for large language models. *Preprint*, arXiv:2306.11695.

MLC team. 2023. MLC-LLM.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. 2023a. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *Preprint*, arXiv:2203.05740.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. 2023b. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. *arXiv preprint arXiv:2304.09145*.

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.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. GLM-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-language models for vision tasks: A survey. *Preprint*, arXiv:2304.00685.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

Xiangguo Zhang, Haotong Qin, Yifu Ding, Ruihao Gong, Qinghua Yan, Renshuai Tao, Yuhang Li, Fengwei Yu, and Xianglong Liu. 2021. Diversifying sample generation for accurate data-free quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15658–15667.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. Swift:a scalable lightweight infrastructure for fine-tuning. *Preprint*, arXiv:2408.05517.

Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Zihan Wang, Lei Shen, Andi Wang, Yang Li, Teng Su, Zhilin Yang, and Jie Tang. 2023. Codegeex: A pre-trained model for code generation with multilingual benchmarking on humaneval-x. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5673–5684.

Content

A Appendix

Technique	Approach	Strategy	Eq. Trans.	Algorithm
TRANSFORMATION	*Rule-based*	$\boldsymbol{s} = \max(\|\boldsymbol{X}\|^{\gamma})/\max(\|\boldsymbol{W}\|^{1-\gamma}), \gamma = 0.5, 0.75, ...$	✓	SmoothQuant(Xiao et al., 2023)
		$\boldsymbol{Q}$, where $\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}$ and $\|\boldsymbol{Q}\| = 1$	✓	QuaRot (Ashkboos et al., 2024)
	Search-based	$\boldsymbol{s} = \max(\|\boldsymbol{X}\|^{\gamma})/\max(\|\boldsymbol{W}\|^{1-\gamma})$, grid search for $\gamma \in [0, 1]$	✓	AWQ(Lin et al., 2023)
		$\boldsymbol{s} = \max(1.0, \max(\boldsymbol{X})/t)$, grid search for t	✓	OS+(Wei et al., 2023b)
	Learnining-based	$\boldsymbol{s} = \arg\min_{\boldsymbol{s}} \mathcal{L}, \boldsymbol{s} \leftarrow \boldsymbol{s} - \eta \frac{\partial \mathcal{L}(\boldsymbol{s})}{\partial \boldsymbol{s}}$	✓	OmniQuant(Shao et al., 2023)
CLIPPING	*Rule-based*	$\alpha = 1, \beta = 1$	✓	SmoothQuant(Xiao et al., 2023), OS+(Wei et al., 2023b), GPTQ(Frantar et al., 2022), QuaRot (Ashkboos et al., 2024)
	Search-based	grid search for $\alpha = \beta \in [0, 1]$	✗	AWQ(Lin et al., 2023)
	Learning-based	$\alpha, \beta = \arg\min_{\alpha,\beta} \mathcal{L}, \alpha \leftarrow \alpha - \eta \frac{\partial \mathcal{L}(\alpha)}{\partial \alpha}, \beta \leftarrow \beta - \eta \frac{\partial \mathcal{L}(\beta)}{\partial \beta}$	✗	OmniQuant(Shao et al., 2023)
RECONSTRUCTION	*Hessian-based*	$\boldsymbol{W} \leftarrow \boldsymbol{W} - \boldsymbol{E}\boldsymbol{H}^{-1}, \boldsymbol{H}^{-1} = \left(2\boldsymbol{X}\boldsymbol{X}^{\top} + \lambda \boldsymbol{I}\right)^{-1}$	✗	GPTQ(Frantar et al., 2022)

Table 13: Detailed comparison of the three main strategies in the main text. **Eq. Trans.** indicates whether the algorithm is an equivalent transformation. γ is the scaling factor. $\boldsymbol{s}$ and $\boldsymbol{Q}$ represent transformation vector and matrix. $\boldsymbol{I}$ is the identity matrix. $\mathcal{L}$ is the loss function with the learning rate η. α and β mean clipping minimum and maximum value. $\boldsymbol{H}$ is Hessian matrix, and $\boldsymbol{E}$ denotes quantization errors calculated with $\boldsymbol{H}$. λ is the decay coefficient.

A.1 Preliminary for Quantization

A complete uniform quantization process can be formulated by:

$$\bar{\boldsymbol{w}} = \text{clip}(\left\lfloor \frac{\boldsymbol{w}}{s} \right\rceil + z, N_{min}, N_{max}), \\ \hat{\boldsymbol{w}} = s \cdot (\bar{\boldsymbol{w}} - z), \tag{1}$$

where $s \in \mathbb{R}_+$ and $z \in \mathbb{Z}$ are called *scale* and *zero-point*, respectively. $\lfloor \cdot \rceil$ rounds the continuous numbers to the nearest integers. Eq. 1 first quantizes the weights or activations into the target integer range $[N_{\min}, N_{\max}]$ and then de-quantizes the integers to the original range.

Naive quantization can be split into four dimensions: bit-width, symmetric/asymmetric, group size, and dynamic/static.

Bit-width: Given t bits, $[N_{\min}, N_{\max}]$ is determined by $[-2^{t-1}, 2^{t-1} - 1]$. In this paper, the notion "wxay" is employed to represent the bit-widths of weights "w" and activations "a";

Symmetric or asymmetric. For asymmetric quantization, a zero-point value z will usually be introduced to represent the floating-point zero. Otherwise, the symmetric quantization does not have that adjustable z to adapt various ranges;

Group size. Shen et al. (2020) first proposes group-wise quantization, which divides each channel of a weight [14] into different groups and employs a different set of scale and zero-point for each group $\boldsymbol{W}_{i,j:j+g}$ with group size g. However, per-tensor ($\boldsymbol{W}_{:,:}$) quantization or per-channel ($\boldsymbol{W}_{i,:}$) quantization can be also seen as group-wise quantization with a larger group size;

Dynamic or static. Due to variance in activation range for LLM, Yao et al. (2022) first introduces token-wise ($\boldsymbol{X}_{i,:}$) quantization for activation, which dynamically calculates the min/max range for each token during model inference. We also measure dynamic/static per-tensor activation quantization to make a comprehensive comparison.

As outlined in Table 13, we also summarize the three strategies, *e.g.*, transformation, clipping, and reconstruction in the main text and define their behavior. Additionally, for the equivalence transformation categories OS+ and OmniQuant, considering that we are using the LLaMA series models (which have layers without bias), we aim to avoid introducing additional computations into the model's inference process. Therefore, we have decided not to explore the shift operation involved in these two methods.

[14] We denote weight $\boldsymbol{W} \in \mathbb{R}^{out \times in}$. The first/second dimension of $\boldsymbol{W}$ represents output/input channels. Notably, we ignore the batch size dimension for activation $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, where n means token number, d means hidden size.

A.2 More Implementation Details

Unless otherwise specified, our implementation adopts asymmetric quantization for both activations and weights. Specifically, we apply per-token dynamic quantization for activations and static quantization for weights. g128 and g64 represent two commonly used settings in group weight quantization, indicating group sizes of 128 and 64, respectively. In line with previous works Shao et al. (2023); Liu et al. (2024); Ashkboos et al. (2024), For OmniQuant, the learning rate for weight clipping and transformation is $5e^{-3}$ and $1e^{-2}$ during the reconstruction phase. We follow the default setting of 20 learning epochs. Besides, we employ the evaluation tool OpenCompass (Contributors, 2023) with LightLLM (ModelTC, 2023) as the backend on Nvidia A100 80G GPU to benchmark downstream tasks. Additionally, we evaluate PPL with 2048 sequence length in our own LLMC.

A.3 PPL Alignment with the Existing Methods

Method	Calib. Data	Sequence Length	Number of Samples	Seed
GPTQ	C4	2048	128	0
AWQ	Pile (val)	512	128	42
Omniquant	Wikitext2	2048	128	2
Smoothquant	Pile(val)	512	128	42
OS+	Pile (val)	512	128	42
Quarot	Wikitext2	2048	128	0
Wanda	pileval	512	512	42

Table 14: Calibration and hyperparameter settings in our alignment experiments.

In this section, we conduct some alignment experiments with several established quantization algorithms (LLMC *vs.* original paper/codes). Our experimental settings are the same as the original paper or default settings of their open-source codes (as shown in Table 14). These experimental results are summarized in Table 15, Table 16, Table 17, and Table 18. The performance from the tables illustrates that our LLMC tool achieves performance almost identical to the original quantization algorithms reported in the literature. By employing these experiments, we demonstrate that our tool is not only effective but also reliable in reproducing the outcomes of existing quantization methods. This ensures that our contributions are both credible and valuable to the ongoing research in LLM quantization.

Method	w4g128	w3g128	w2g64
GPTQ	5.62	6.32	14.97
GPTQ-LLMC	5.62	6.32	14.97
AWQ	5.60	6.24	2.16e5
AWQ-LLMC	5.60	6.24	2.16e5
OmniQuant	5.59	6.09	9.53
OmniQuant-LLMC	5.59	6.09	9.53

Table 15: Wikitext2 PPL alignment results of weight-only asymmetric quantization of LLaMA-2-7B Model. "-LLMC" means our implementation with the LLMC toolkit.

Method	w8a8	w6a6	w4a4
OmniQuant	5.49	5.70	12.21
OmniQuant-LLMC	5.49	5.70	12.23
Quarot *w/* GPTQ.	5.48	5.50	6.22
Quarot-LLMC *w/* GPTQ-LLMC.	5.48	5.50	6.24

Table 16: Wikitext2 PPL alignment results of weight-activation asymmetric quantization of LLaMA-2-7B Model.

Method	LLaMA-2-7b	LLaMA-2-70b	LLaMA-3-8b	LLaMA-3-70b
Wanda	6.91	4.22	9.56	OOM
Wanda-LLMC	6.91	4.19	9.58	5.75

Table 17: Wikitext2 PPL alignment results of 50% unstructured sparsification method Wanda (Sun et al., 2024) for LLaMA-2-7B, 70B, and LLaMA-3 family.

Method	w8a8
SmoothQuant	5.589
SmoothQuant-LLMC	5.589
OS+	5.511
OS+-LLMC	5.517

Table 18: Wikitext2 PPL alignment results of weight-activation symmetric quantization of LLaMA-2-7B Model.

Model	KV Cache Prec.	Pass@1 (%) ↑		
		Human-Eval	MBPP	Avg.
LLaMA-2-7B	Full Prec.	12.80	22.00	17.40
	int8	13.41	20.00	16.71
	int4	13.41	21.00	17.21
	int2	0.00	0.00	0.00
	w4a8kv4	12.20	18.40	15.30
LLaMA-2-13B	Full Prec.	18.29	24.00	21.15
	int8	17.68	23.00	20.34
	int4	17.68	23.00	20.34
	int2	0.00	0.00	0.00
	w4a8kv4	15.85	23.40	19.63
LLaMA-2-70B	Full Prec.	29.27	42.00	35.64
	int8	29.88	38.00	33.94
	int4	30.49	39.00	34.75
	int2	0.00	0.00	0.00
	w4a8kv4	29.27	38.20	33.74

Table 19: Naive KV cache quantization results on Human-Eval and MBPP for LLAMA-2 series models. We employ group-wise quantization (*i.e.*, g8) here.

A.4 KV Cache Quantization

This part shows the accuracy of KV cache quantization for code generation tasks. From Table 19, we can find that the naive int8 and int4 KV cache quantization brings almost no accuracy degradation for both the Human-Eval and MBPP datasets. This conclusion proves that the naive 4-bit KV cache can be adopted without harm to performance. However, the naive 2-bit KV cache will bring a crash for the generation, and thus should not be adopted. Similar results can be found in Table 23 for long-context evaluation.

A.5 Extensibility of LLMC

To further demonstrate the extensibility of the toolkit, we conduct extensive experiments, including MoE quantization (shown in Table 20), VLM quantization (shown in Table 21), and sparsification (shown in Table 24).

MOE quantization. We utilize our toolkit to evaluate the performance of quantized Mixtral-8x7B, as shown in Table 20.

#Bits	Method	PPL ↓		
		WikiText2	C4	Avg.
Full Prec.	-	3.84	7.40	5.62
w4a16g128	AWQ	4.05	7.59	5.82
	GPTQ	4.05	7.60	5.82
w3a16g128	AWQ	4.73	8.29	7.07
	GPTQ	4.93	8.52	7.18
w8a8	SmoothQuant	3.87	7.48	5.68
	OS+	3.87	7.48	5.68
w6a6	SmoothQuant	4.28	7.89	6.09
	OS+	4.27	7.90	6.09

Table 20: Ablation results of Mixtral-8x7B weight-only quantization and weight-activation quantization.

VLM quantization. For the VLM quantization, the quantized LLaVA-7B is evaluated by our toolkit on Perception and Cognition tasks, as depicted in Table 21.

Sparsity. Table 24 presents the results for the LLaMA-2-7B, 70B, and LLaMA-3 family of models obtained using the sparsification method Wanda Sun et al. (2023).

Mixed precision. Table 22 presents the results for weight-only mixed precison on LLaMA-2-7B and LLaMA-3-8B. Mixed precision is an effective method for mitigating quantization errors. More than specific algorithms, LLMC also supports customized layer-wise bit allocation. We found that

#Bits	Method	PPL ↓		
		Perception	Cognition	Avg.
Full Prec.	-	1477.60	283.21	880.40
w4a16g128	AWQ	1441.85	276.78	859.31
	GPTQ	1416.23	285.0	850.61
w3a16g128	AWQ	1417.28	259.64	838.46
	GPTQ	1346.07	280.71	813.39
w8a8	SmoothQuant	1468.93	281.07	875.0
	OS+	1467.28	280.71	873.99
w6a6	SmoothQuant	1469.67	298.21	883.94
	OS+	1467.20	299.64	883.42

Table 21: Ablation results of LLaVA-7B weight-only quantization and weight-activation quantization.

5-bit to 8-bit precision for the `down_proj` offer almost the same benefits.

	LLaMA-2-7B	LLaMA-3-8B
Full Prec.	5.47	6.14
w3a16g128	6.16	8.08
w3a16g128 *w/* `down_proj`-w8a16g128	5.93	7.45
w3a16g128 *w/* `down_proj`-w6a16g128	5.94	7.44
w3a16g128 *w/* `down_proj`-w5a16g128	5.95	7.48
w3a16g128 *w/* `down_proj`-w4a16g128	5.99	7.61

Table 22: PPL results on Wikitext2 of mixed precision with AWQ. We only apply higher bit allocation for `down_proj`, as it vastly impacts the performance mentioned in the main text.

A.6 Inference Speed

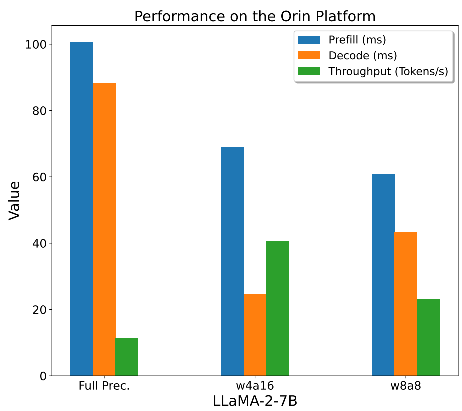

Figure 6: Throughput comparison of quantization on the edge GPU (Drive Orin). (Token/s)

To assess the practical benefits of different quantization approaches, we conducted evaluations [15] using NVIDIA's cloud (SMX 80G A100) and edge (Drive Orin) GPUs, alongside the official inference library, TensorRT-LLM (Nvidia, 2023). Part of our results, as depicted in Figure 9, highlight the throughput improvements achieved for models with 32,000 input tokens and 512 output tokens. The findings indicate that quantization with 8-bit weights and activations enhances the prefill stage's speed by 20%-30% and the decode stage by 40%-60%. In contrast, 4-bit weight-only quantization reduces the prefill speed by 10% but increases the decode speed by 40%-60%. It's important to note that these acceleration rates tend to diminish for larger models. Besides, 8-bit KV cache quantization has minimal impact on prefill times and slightly reduces decoding throughput for very large models, such as those with 70B models. Figure 7 and Figure 8 supplementarily illustrated the speedup brought by various quantization schemes on 1K and 4K input context length. We can also find that the conclusion for these two scenarios is the same as the 32K input context length. Moreover, Figure 6 shows the speed up on the Drive Orin edge GPU. It can be seen that weight-only quantization also helps the prefill under this setting, which is different from cloud GPUs.

A.7 Detailed Accuracy & PPL

This section presents detailed data from some of the experiments discussed in the main text. Table 25 and Table 26 shows the detailed data for Table 4. Table 27 shows the detailed data for Table 6.

A.8 Results for Various Model Families

From Table 28 to Table 34, we report quantization results for different model families, including SmolLM [16], MiniCPM (Hu et al., 2024), and Qwen2 (Yang et al., 2024). We additionally provide the results on SIQA (Sap et al., 2019), ARC-c (Clark et al., 2018), OBQA (Luo et al., 2021), and WinoGrande (Sakaguchi et al., 2019).

[15] In this section, all weight-only quantization employ 128g group-wise quantization.

[16] https://huggingface.co/blog/smollm

Model	KV Cache Prec.	Accuracy (%) ↑				
		NarrativeQA	QASPER	MultiFieldQA-en	MultiFieldQA-zh	Avg.
ChatGLM3-6B-32k	Full Prec.	25.93	43.35	51.57	62.36	45.80
	int8	25.74	43.57	51.81	62.48	45.90
	int4	26.13	43.43	51.63	61.04	45.56
	int2	1.89	4.68	3.13	1.08	2.70

Table 23: KV cache quantization results on Single-Document QA from LongBench (Bai et al., 2023)

Model	Sparsity							
	Dense		25%		50%		75%	
	C4	Wikitext2	C4	Wikitext2	C4	Wikitext2	C4	Wikitext2
LLaMa2-7B	7.26	5.47	7.46	5.61	9.25	6.85	260.42	259.91
LLaMa2-70B	5.71	3.32	5.76	3.4	6.49	4.17	32.5	21.66
LLaMa3-8B	9.44	6.13	10.01	6.47	15.07	9.68	336.62	290.38
LLaMa3-70B	7.16	2.85	7.44	3.22	9.96	5.81	93.99	74.78

Table 24: Perplexity results of LLaMA-2-7B, 70B, and LLaMA-3 family under Wanda method.

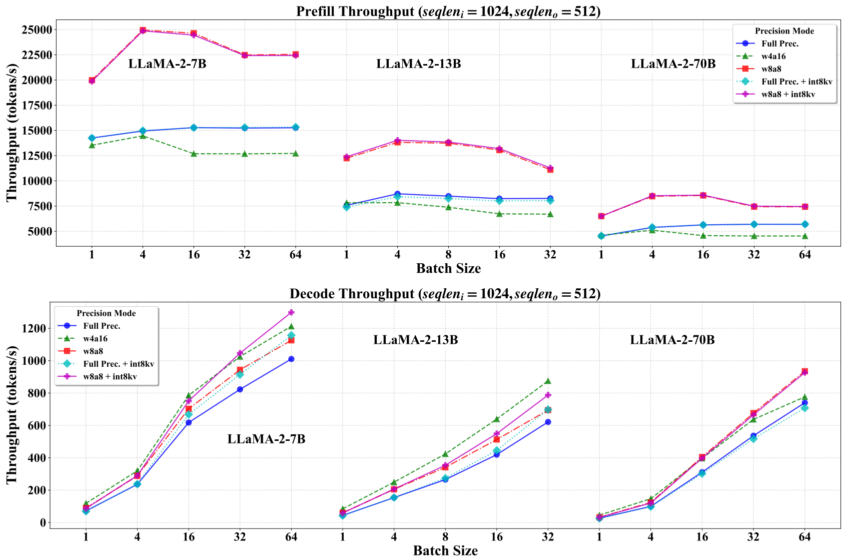

Figure 7: Inference speed of 7B, 13B, and 70B LLaMA-2 models on NVIDIA A100 GPU. (Input sequence length: 1024, Output sequence length: 512)

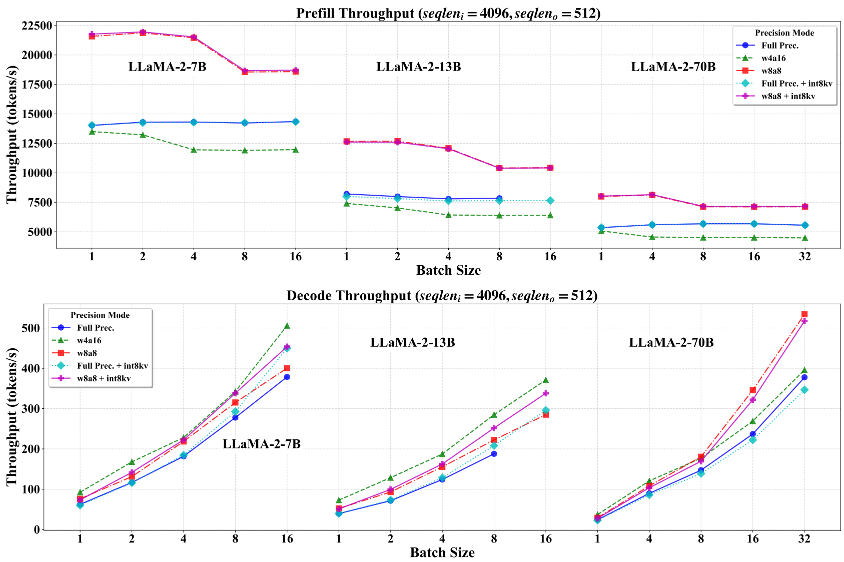

Figure 8: Inference speed of 7B, 13B, and 70B LLaMA-2 models on NVIDIA A100 GPU. (Input sequence length: 4096, Output sequence length: 512)

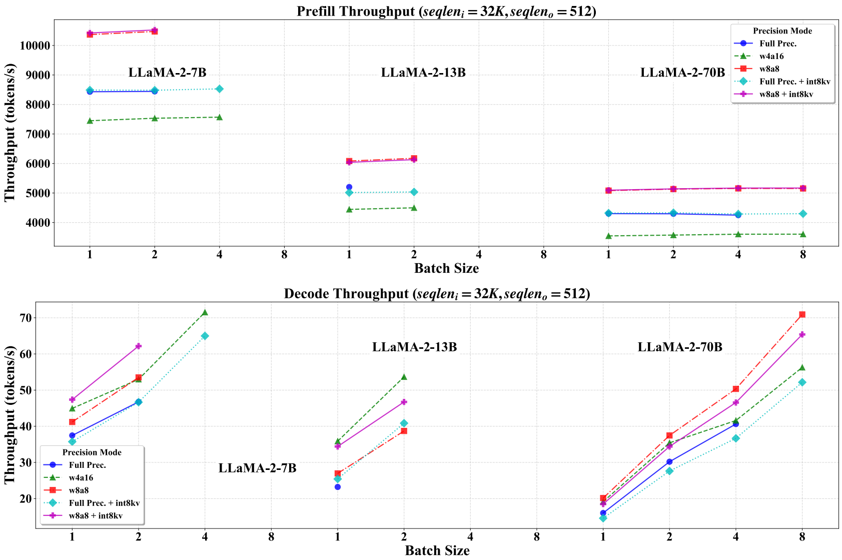

Figure 9: Inference speed of 7B, 13B, and 70B LLaMA-2 models on NVIDIA A100 GPU. (Input sequence length: 32K, Output sequence length: 512)

#Bits	Method	PPL ↓			Accuracy (%) ↑					
		WikiText2	C4	Avg.	MMLU	ARC-e	BoolQ	HellaSwag	PIQA	Avg.
w3a16g128	AWQ	6.22	8.28	7.25	38.10	48.56	71.78	70.86	76.61	61.18
	AWQ *w/* asym. clip	6.18	8.24	7.21	42.33	47.09	71.44	70.93	76.17	61.59
w2a16g64	AWQ	2.09e5	1.59e5	1.8e5	25.38	4.87	62.17	24.83	51.2	37.69
	AWQ *w/* asym. clip	11.69	14.83	13.26	27.4	25.4	63.27	57.4	70.4	48.77

Table 25: Results of asymmetric/symmetric weight clipping for LLaMA-2-7B model.

#Bits	Method	PPL ↓			Accuracy (%) ↑					
		WikiText2	C4	Avg.	MMLU	ARC-e	BoolQ	HellaSwag	PIQA	Avg.
w3a16g128	AWQ	3.75	6.05	4.90	67.54	87.65	86.57	81.11	81.88	80.95
	AWQ *w/* asym. clip	3.74	6.04	4.89	67.07	89.95	86.30	80.95	81.07	81.07
w2a16g64	AWQ	7.1e4	6.5e4	6.8e4	24.46	26.46	37.83	24.60	50.87	32.84
	AWQ *w/* asym. clip	5.24	7.73	6.49	57.91	80.07	83.91	75.98	78.67	75.31

Table 26: Results of asymmetric/symmetric weight clipping for LLaMA-2-70B model.

#Bits	Method	PPL ↓			Accuracy (%) ↑					
		WikiText2	C4	Avg.	MMLU	ARC-e	BoolQ	HellaSwag	PIQA	Avg.
w3a16g128	GPTQ	8.28	13.07	10.67	57.81	78.48	73.49	72.16	77.86	71.96
	AWQ	8.57	13.39	10.98	54.35	74.78	74.56	71.85	78.07	70.72
	AWQ *w/* GPTQ	8.18	12.91	10.55	59.10	80.60	73.12	72.40	78.40	72.72
	Quarot	40.81	59.20	50.00	29.03	29.98	58.87	45.18	66.43	45.90
	Quarot *w/* GPTQ	7.99	12.70	10.35	60.25	83.25	78.56	72.96	79.16	74.84

Table 27: Results of reconstruction (GPTQ) combined with scaling (AWQ) and rotation-based (QuaRot) transformation for LLaMA-3-8B model. Clipping for AWQ here is canceled to expel distractions.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	17.56	22.17	19.86	53.04	39.51	68.34	23.00	35.14	60.00	61.36	25.94	45.79
w2a16g128	RTN	2.27e+07	3.06e+07	2.66e+07	51.38	34.19	**52.61**	**18.80**	25.84	**47.74**	24.66	**21.33**	**34.57**
	GPTQ	1.30e+04	1.04e+04	1.17e+04	52.57	33.16	50.98	16.80	**25.94**	45.26	**27.86**	20.82	34.17
	AWQ	**1.02e+04**	**8.18e+03**	**9.18e+03**	48.93	**34.44**	51.03	15.60	25.62	38.84	26.30	20.48	32.65
w3a16g128	RTN	91.65	96.75	94.20	48.38	36.59	60.88	19.00	30.21	49.54	46.84	21.93	39.17
	GPTQ	**32.89**	**40.29**	**36.59**	51.93	37.15	61.53	**20.80**	**31.39**	**58.56**	**51.89**	22.53	**41.97**
	AWQ	54.20	55.94	55.07	50.36	**37.41**	**62.40**	17.00	31.28	52.23	51.56	**24.49**	40.84
w4a16g128	RTN	22.54	28.04	25.29	53.67	38.54	66.38	**23.00**	**34.18**	**62.05**	58.04	**25.85**	**45.21**
	GPTQ	**20.03**	**25.01**	**22.52**	52.01	**39.71**	65.56	22.00	33.99	56.36	**58.84**	24.74	44.15
	AWQ	21.42	26.19	23.81	52.25	38.02	**66.76**	22.80	34.07	58.96	58.54	25.77	44.65
w4a4	RTN	2.60e+03	2.22e+03	2.41e+03	50.91	33.73	52.61	17.40	26.40	43.73	30.77	18.77	34.29
	SmoothQuant	331.70	441.95	386.82	52.09	33.32	53.70	**18.20**	27.38	**44.46**	37.42	20.65	35.90
	OS+	**263.76**	**389.67**	**326.71**	52.49	**35.62**	55.39	14.60	**27.56**	43.46	41.46	**20.73**	**36.41**
	QuaRot	472.15	567.85	520.00	49.17	34.34	**56.37**	14.60	27.08	41.01	**43.01**	**20.73**	35.79
w6a6	RTN	22.84	27.45	25.14	49.41	38.28	65.07	20.00	33.02	58.23	56.73	**25.68**	43.30
	SmoothQuant	20.37	25.12	22.74	53.91	38.13	64.64	**22.80**	32.52	59.02	59.22	25.00	44.41
	OS+	**19.67**	**25.00**	**22.33**	51.54	**39.71**	**66.81**	21.20	32.88	**59.85**	**60.19**	24.32	44.56
	QuaRot	20.26	25.02	22.64	52.25	39.05	66.32	22.40	**33.06**	57.77	60.14	**25.68**	**44.58**
w8a8	RTN	17.75	22.45	20.10	52.57	39.05	**68.01**	21.80	35.07	**60.37**	61.45	25.09	45.43
	SmoothQuant	17.68	22.35	20.01	52.64	**39.66**	67.74	21.80	**35.14**	60.15	61.49	25.43	45.51
	OS+	**17.67**	**22.32**	**19.99**	53.51	39.00	67.79	**23.00**	**35.14**	60.09	**61.66**	**25.85**	**45.76**
	QuaRot	17.77	22.42	20.10	52.33	39.15	**68.01**	22.80	**35.14**	60.34	61.15	25.34	45.53

Table 28: Quantization Results for SmolLM-135M model. Activation clipping and online rotation within QuaRot are canceled for a fair comparison. "HellaS." and "WinoG." represent HellaSwag and WinoGrande, respectively. We mark the best results in **bold**.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	13.10	17.68	15.39	58.25	41.25	71.33	25.20	41.63	55.20	69.82	33.28	49.49
w2a16g128	RTN	2.98e+06	2.60e+06	2.79e+06	51.22	32.91	51.74	**16.40**	25.72	**47.95**	25.21	**20.90**	**34.01**
	GPTQ	**797.15**	**812.25**	**804.70**	48.62	**34.95**	50.22	16.00	26.24	39.30	27.69	18.69	32.71
	AWQ	3.12e+03	2.67e+03	2.90e+03	48.93	34.08	**52.50**	15.20	**26.82**	42.11	**30.68**	19.97	33.79
w3a16g128	RTN	32.13	39.52	35.83	53.35	36.80	**67.30**	**22.20**	36.23	**62.02**	57.87	29.44	**45.65**
	GPTQ	**21.14**	**26.85**	**24.00**	52.64	37.77	65.56	19.40	36.47	51.96	57.95	27.99	43.72
	AWQ	23.24	28.91	26.08	53.75	**38.28**	66.76	21.00	**37.86**	53.91	**61.41**	**29.86**	45.35
w4a16g128	RTN	15.11	20.20	17.65	56.20	40.53	**70.46**	**24.20**	40.39	54.37	65.87	32.00	48.00
	GPTQ	**14.80**	**19.72**	**17.26**	55.72	39.36	69.91	23.80	39.75	**54.43**	66.20	31.14	47.54
	AWQ	15.17	20.08	17.63	57.06	**40.94**	69.26	23.00	**41.00**	51.74	**68.27**	**32.85**	**48.02**
w4a4	RTN	645.64	613.99	629.82	51.14	33.88	54.52	13.80	26.51	43.61	33.96	19.37	34.60
	SmoothQuant	123.40	233.90	178.65	48.70	35.36	**59.47**	**17.20**	30.35	45.17	44.87	**24.66**	**38.22**
	OS+	**80.14**	**122.98**	**101.56**	49.96	**35.41**	58.43	13.20	**30.46**	**47.06**	**48.70**	21.67	38.11
	QuaRot	157.89	158.13	158.01	49.41	34.44	57.73	15.80	28.28	39.08	40.57	21.16	35.81
w6a6	RTN	15.32	21.15	18.24	55.17	40.23	69.15	23.00	39.44	48.78	66.46	30.89	46.64
	SmoothQuant	14.26	19.17	16.72	53.20	40.99	69.53	**26.80**	40.84	53.98	67.85	32.08	**48.16**
	OS+	**14.15**	**19.01**	**16.58**	54.14	**41.40**	**69.75**	23.00	**40.86**	53.88	67.34	**32.42**	47.85
	QuaRot	14.36	19.24	16.80	54.30	40.84	69.64	24.40	40.41	**55.05**	**68.35**	32.00	48.12
w8a8	RTN	13.31	17.97	15.64	56.04	40.58	70.67	25.80	41.64	55.20	**70.24**	33.79	49.24
	SmoothQuant	13.27	17.90	15.58	56.75	**41.30**	70.95	25.80	41.67	**55.96**	70.03	33.53	49.50
	OS+	**13.24**	**17.85**	**15.55**	55.96	41.10	**71.16**	**26.20**	41.67	55.84	70.16	**34.04**	**49.52**
	QuaRot	13.26	17.90	15.58	56.75	40.89	**71.16**	25.20	**41.73**	53.82	69.87	33.87	49.16

Table 29: Quantization Results for SmolLM-350M model.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	9.58	13.92	11.75	60.93	43.65	75.79	30.00	49.55	65.93	76.47	43.43	55.72
w2a16g128	RTN	1.40e+07	1.06e+07	1.23e+07	49.64	33.42	53.10	**17.20**	25.85	44.50	25.42	22.61	33.97
	GPTQ	465.98	319.93	392.95	51.70	34.60	51.25	15.60	27.03	51.38	30.68	19.28	35.19
	AWQ	**91.93**	**122.20**	**107.06**	49.64	**34.65**	**60.72**	16.40	**31.11**	**56.36**	**50.38**	**23.38**	**40.33**
w3a16g128	RTN	17.57	23.43	20.50	56.99	41.20	72.36	**28.60**	**45.72**	61.47	70.20	**39.93**	52.06
	GPTQ	**12.10**	16.85	14.47	58.56	40.89	73.01	27.80	45.21	61.56	71.09	37.37	51.94
	AWQ	12.11	**16.68**	**14.40**	57.70	**41.81**	**73.34**	28.20	45.22	**63.91**	**72.81**	39.76	**52.84**
w4a16g128	RTN	10.56	15.13	12.85	60.30	**44.52**	75.08	**31.20**	**49.12**	63.00	**76.05**	**43.52**	**55.35**
	GPTQ	**10.05**	14.45	12.25	60.54	43.76	74.97	29.40	48.43	65.29	75.67	42.41	55.06
	AWQ	10.05	**14.43**	**12.24**	60.77	43.50	**75.79**	29.60	48.56	**65.57**	75.97	42.92	55.34
w4a4	RTN	1.34e+07	8.32e+07	4.83e+07	50.59	33.06	50.98	14.80	24.50	48.87	29.38	22.18	34.30
	SmoothQuant	285.34	222.59	253.96	51.62	34.24	54.46	15.60	29.47	55.78	42.68	23.29	38.39
	OS+	403.41	882.42	642.91	47.99	36.03	55.77	17.40	29.64	54.04	47.60	25.00	39.18
	QuaRot	**37.41**	**49.55**	**43.48**	50.20	**37.15**	**60.07**	**17.80**	**34.05**	**58.90**	**52.10**	**26.45**	**42.09**
w6a6	RTN	11.71	16.65	14.18	56.20	41.97	73.29	28.60	46.47	63.73	72.81	38.40	52.68
	SmoothQuant	10.71	15.54	13.12	59.35	42.27	**74.43**	**30.40**	47.87	64.46	74.37	39.85	54.12
	OS+	10.51	15.13	12.82	58.96	**42.43**	73.99	29.20	48.25	64.83	73.78	40.44	53.98
	QuaRot	**10.35**	**14.99**	**12.67**	58.09	**42.43**	73.83	29.60	**48.65**	**65.14**	**74.66**	**40.70**	**54.14**
w8a8	RTN	9.73	14.21	11.97	59.67	**43.86**	**76.01**	**30.60**	49.40	66.02	76.35	42.92	55.60
	SmoothQuant	9.65	14.04	11.84	61.33	43.50	75.63	30.40	49.37	65.81	76.60	43.00	55.71
	OS+	9.64	14.01	11.83	60.46	43.65	75.63	30.00	**49.43**	**66.36**	**76.73**	**44.03**	**55.79**
	QuaRot	**9.64**	**14.01**	**11.82**	59.91	43.30	75.79	30.20	49.33	**66.36**	76.64	43.43	55.62

Table 30: Quantization Results for SmolLM-1.7B model.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	8.60	13.74	11.17	60.62	45.04	74.48	23.20	50.09	68.23	70.37	36.26	53.54
w2a16g128	RTN	7.86e+03	1.61e+04	1.20e+04	50.91	34.54	53.10	13.80	25.90	40.70	26.39	22.44	33.47
	GPTQ	**71.23**	**101.64**	**86.44**	48.86	36.03	57.24	16.20	29.00	**43.12**	33.88	19.45	35.47
	AWQ	100.70	197.93	149.31	52.64	**38.18**	**60.83**	**16.60**	**31.88**	42.60	**44.78**	**22.95**	**38.81**
w3a16g128	RTN	11.00	17.70	14.35	60.77	41.50	72.42	19.60	46.76	63.79	63.93	33.28	50.26
	GPTQ	10.34	16.44	13.39	60.62	42.99	71.60	21.80	46.40	60.64	65.11	**35.24**	50.55
	AWQ	**10.01**	**16.23**	**13.12**	59.67	**44.52**	**72.63**	**22.40**	**47.07**	**65.38**	**66.84**	34.30	**51.60**
w4a16g128	RTN	8.98	14.35	11.67	59.43	44.37	73.07	23.20	**49.42**	67.13	**69.40**	36.35	52.80
	GPTQ	8.89	**14.23**	**11.56**	60.46	44.06	**73.39**	22.80	49.00	69.24	68.64	36.09	52.96
	AWQ	**8.87**	14.25	11.56	61.17	**45.19**	73.29	**23.40**	49.36	**71.01**	69.32	**36.60**	**53.67**
w4a4	RTN	35.70	50.17	42.93	52.17	39.05	64.09	16.60	36.29	57.34	51.47	25.51	42.81
	SmoothQuant	19.75	30.51	25.13	52.33	40.48	65.45	19.00	**40.68**	60.40	**55.18**	28.07	**45.20**
	OS+	21.72	33.72	27.72	51.22	**40.79**	**65.67**	**20.20**	40.59	59.82	53.79	**28.67**	45.09
	QuaRot	**19.18**	**30.01**	**24.60**	52.01	35.31	59.30	18.00	28.77	**63.39**	41.25	26.54	40.57
w6a6	RTN	9.09	14.44	11.77	61.01	**44.58**	**74.16**	22.40	49.33	69.30	69.11	**36.60**	**53.31**
	SmoothQuant	9.03	14.39	11.71	60.06	44.11	73.18	**23.40**	49.25	69.24	**69.70**	36.09	53.13
	OS+	9.05	**14.38**	11.72	59.83	44.52	73.88	**23.40**	**49.48**	68.93	68.94	36.01	53.12
	QuaRot	**9.01**	14.41	**11.71**	58.80	36.85	65.29	20.20	31.16	**69.48**	47.35	29.35	44.81
w8a8	RTN	8.65	13.80	11.23	62.04	44.17	**74.48**	**23.80**	49.86	68.10	70.08	**36.52**	**53.63**
	SmoothQuant	8.64	13.79	11.21	59.91	44.11	74.32	22.20	49.90	68.32	**70.29**	35.67	53.09
	OS+	**8.63**	**13.78**	**11.21**	59.91	**44.63**	74.16	23.00	**49.92**	68.17	70.08	35.67	53.19
	QuaRot	8.64	13.79	11.22	60.38	37.15	64.96	22.40	31.53	**68.99**	48.06	29.61	45.38

Table 31: Quantization Results for MiniCPM-1B model.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	8.16	13.00	10.58	63.14	47.24	76.22	28.60	52.88	73.58	74.66	42.58	57.36
w2a16g128	RTN	612.79	880.31	746.55	49.01	35.52	56.64	15.80	28.51	58.93	31.86	20.14	37.05
	GPTQ	29.60	45.30	37.45	47.75	36.44	60.88	15.20	32.90	55.87	38.85	21.67	38.69
	AWQ	**24.28**	**36.25**	**30.26**	55.09	**40.07**	**66.10**	**16.80**	**39.54**	**63.70**	**55.89**	**29.35**	**45.82**
w3a16g128	RTN	9.79	15.54	12.66	60.22	44.58	**74.48**	25.80	50.42	71.83	70.12	**40.78**	54.78
	GPTQ	9.56	15.29	12.43	61.33	43.91	73.50	25.40	50.23	73.12	69.74	37.97	54.40
	AWQ	**9.18**	**14.68**	**11.93**	60.85	**46.21**	74.05	**27.20**	**51.07**	**73.36**	**71.76**	40.27	**55.60**
w4a16g128	RTN	8.40	13.43	10.92	64.96	47.34	**76.22**	**28.80**	52.77	73.70	74.45	**42.24**	**57.56**
	GPTQ	8.50	13.59	11.04	61.88	**47.39**	75.30	27.40	52.65	**75.14**	73.40	41.89	56.88
	AWQ	**8.32**	**13.39**	**10.85**	61.33	46.88	75.73	**28.80**	**52.80**	74.65	**74.96**	41.72	57.11
w4a4	RTN	33.64	52.72	43.18	53.35	38.64	64.85	18.00	37.21	62.60	52.23	26.71	44.20
	SmoothQuant	17.20	**28.01**	**22.61**	53.99	**41.91**	68.39	**23.20**	42.71	**63.88**	**60.02**	**33.28**	**48.42**
	OS+	**17.15**	28.22	22.68	53.75	41.15	**68.50**	20.40	**43.25**	63.82	59.22	31.91	47.75
	QuaRot	19.87	31.97	25.92	53.51	35.98	61.04	16.80	27.36	61.50	40.91	25.09	40.27
w6a6	RTN	8.46	13.58	11.02	63.14	**45.96**	75.03	27.60	**52.21**	73.21	73.78	40.61	56.44
	SmoothQuant	**8.43**	**13.49**	**10.96**	62.59	45.24	**75.63**	27.80	52.03	72.75	73.99	41.21	56.41
	OS+	8.45	13.51	10.98	61.33	45.55	74.65	**28.00**	52.01	**74.07**	**74.16**	**41.81**	**56.45**
	QuaRot	8.48	13.55	11.01	61.56	38.33	65.18	20.60	28.49	72.23	52.36	31.91	46.33
w8a8	RTN	**8.13**	13.04	**10.59**	63.77	46.57	**76.39**	29.40	**52.97**	73.94	**74.66**	42.24	**57.49**
	SmoothQuant	8.17	13.04	10.60	63.06	46.93	76.33	29.20	52.80	73.73	74.62	**42.32**	57.37
	OS+	8.18	**13.04**	10.61	63.06	**47.19**	76.17	**29.60**	52.80	**74.01**	74.28	41.89	57.38
	QuaRot	8.18	13.04	10.61	62.75	38.43	66.05	22.40	28.57	73.58	53.07	31.83	47.09

Table 32: Quantization Results for MiniCPM-2B model.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	13.58	18.97	16.27	57.70	43.04	69.48	21.40	38.35	61.04	54.76	25.51	46.41
w2a16g128	RTN	2.09e+05	1.97e+05	2.03e+05	51.30	34.03	53.37	14.40	25.48	44.86	25.00	**22.70**	33.89
	GPTQ	**1.34e+03**	**1.39e+03**	**1.37e+03**	50.04	**34.49**	**53.70**	13.80	25.95	44.28	27.86	20.90	33.88
	AWQ	9.73e+03	8.82e+03	9.27e+03	48.54	33.06	53.37	**15.00**	**26.30**	**46.36**	**28.75**	20.14	**33.94**
w3a16g128	RTN	32.82	45.18	39.00	52.64	37.51	62.62	18.80	33.34	45.08	46.04	23.38	39.93
	GPTQ	**19.62**	**28.06**	**23.84**	52.96	37.10	**66.43**	**19.00**	34.71	**59.60**	**51.98**	**24.49**	**43.28**
	AWQ	22.72	30.28	26.50	52.96	**39.00**	66.16	18.00	**35.18**	57.34	49.28	23.72	42.70
w4a16g128	RTN	15.75	21.90	18.83	54.54	40.69	67.68	**21.20**	37.42	**62.32**	51.01	23.98	44.86
	GPTQ	**14.86**	**20.80**	**17.83**	55.49	41.04	67.90	21.00	37.47	59.17	**56.86**	24.57	**45.44**
	AWQ	14.90	20.86	17.88	56.99	**41.20**	**68.44**	19.20	**37.50**	59.45	52.90	**24.83**	45.06
w4a4	RTN	1.09e+03	1.01e+03	1.05e+03	48.86	34.60	52.45	13.20	26.23	41.71	27.86	19.62	33.07
	SmoothQuant	172.65	232.83	202.74	49.72	34.49	54.73	12.80	27.93	45.66	32.58	21.33	34.90
	OS+	261.88	271.76	266.82	52.09	33.93	56.75	15.20	28.44	46.02	33.84	21.33	35.95
	QuaRot	**57.48**	**78.85**	**68.16**	51.54	**35.21**	**59.63**	**15.80**	**30.25**	**48.20**	**38.97**	**21.42**	**37.63**
w6a6	RTN	15.79	21.99	18.89	53.99	40.33	67.08	21.00	37.14	50.46	53.66	26.37	43.75
	SmoothQuant	15.29	21.25	18.27	55.09	41.04	67.19	**21.40**	37.63	54.34	53.37	**26.54**	44.58
	OS+	15.32	21.22	18.27	54.78	**42.07**	**68.44**	20.40	**37.92**	53.33	54.76	25.85	44.69
	QuaRot	**14.93**	**20.82**	**17.87**	55.17	41.56	67.63	**21.40**	37.62	**57.40**	**55.72**	25.43	**45.24**
w8a8	RTN	13.85	19.37	16.61	56.12	42.22	69.37	**21.80**	**38.32**	58.93	54.59	25.17	45.81
	SmoothQuant	13.72	19.20	16.46	56.99	42.37	**69.80**	21.00	38.29	59.97	54.71	25.60	46.09
	OS+	**13.70**	**19.16**	**16.43**	58.33	**42.73**	**69.80**	21.20	38.29	59.79	**55.51**	**25.85**	**46.44**
	QuaRot	13.70	19.17	16.44	55.88	42.48	69.64	**21.80**	38.26	**60.61**	55.22	25.26	46.14

Table 33: Quantization Results for Qwen2-0.5B model.

#Bits	Method	PPL ↓			Accuracy (%) ↑								
		WikiText2	C4	Avg.	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	WinoG.	Avg.
Full Prec.	-	9.84	14.36	12.10	64.72	46.11	75.57	26.80	48.31	71.96	65.87	33.45	54.10
w2a16g128	RTN	3.41e+04	2.45e+04	2.93e+04	50.20	32.75	50.60	**15.00**	25.84	**46.12**	25.29	20.48	33.28
	GPTQ	482.42	462.96	472.69	52.49	34.08	54.30	14.60	26.58	44.40	28.96	**20.82**	34.53
	AWQ	**326.04**	**398.14**	**362.09**	51.38	**34.95**	**55.98**	14.80	**28.12**	42.72	**34.72**	20.31	**35.37**
w3a16g128	RTN	15.24	21.27	18.26	61.72	43.19	70.95	23.00	43.65	68.01	60.14	**32.00**	50.33
	GPTQ	**12.39**	**18.54**	**15.47**	61.25	43.24	**71.98**	**25.20**	**44.39**	**68.44**	**62.50**	30.89	**50.99**
	AWQ	13.47	19.40	16.43	62.04	**43.45**	71.22	23.20	44.11	65.96	59.55	28.07	49.70
w4a16g128	RTN	10.59	15.29	12.94	64.01	44.73	74.59	**26.40**	47.21	**72.39**	62.46	31.48	52.91
	GPTQ	**10.28**	**15.02**	**12.65**	66.14	45.29	74.54	**26.40**	**47.68**	71.07	65.07	**32.68**	53.61
	AWQ	10.41	15.16	12.79	66.69	**46.57**	**75.24**	26.00	47.22	70.55	**65.40**	31.83	**53.69**
w4a4	RTN	275.87	265.84	270.85	50.99	34.54	55.77	13.60	28.63	44.68	31.48	20.56	35.03
	SmoothQuant	85.82	105.29	95.56	48.93	35.16	**59.52**	16.60	32.30	45.60	37.29	**23.98**	37.42
	OS+	98.76	115.03	106.89	50.67	**37.10**	56.96	13.00	31.65	46.79	36.41	21.42	36.75
	QuaRot	**42.19**	**56.01**	**49.10**	52.17	35.82	58.65	**17.80**	**34.72**	**50.86**	**38.38**	21.50	**38.74**
w6a6	RTN	11.02	15.83	13.42	63.93	43.91	72.80	25.80	46.91	63.64	62.88	31.83	51.46
	SmoothQuant	10.94	15.74	13.34	63.30	44.83	73.12	25.80	**47.41**	65.57	63.80	32.42	52.03
	OS+	**10.84**	**15.59**	**13.22**	63.77	45.14	73.18	**27.40**	47.27	62.35	62.25	32.25	51.70
	QuaRot	10.86	15.61	13.24	64.17	**46.37**	**74.21**	26.60	47.21	**67.52**	**65.28**	**34.13**	**53.19**
w8a8	RTN	9.96	14.43	12.19	64.88	46.37	75.30	26.80	**48.13**	72.26	65.87	33.11	54.09
	SmoothQuant	9.97	14.41	12.19	65.67	**47.13**	**75.35**	**27.40**	47.98	72.20	**67.34**	**33.19**	**54.53**
	OS+	9.93	14.31	12.12	65.82	46.88	**75.35**	26.40	**48.13**	**72.42**	65.53	**33.19**	54.22
	QuaRot	**9.89**	**14.31**	**12.10**	65.59	46.06	75.03	26.60	48.09	71.65	65.87	33.02	53.99

Table 34: Quantization Results for Qwen2-1.5B model.