

End-to-End On-Device Quantization-Aware Training for LLMs at Inference Cost

Qitao Tan¹ Xiaoying Song² Ningxi Cheng¹ Guoming Li¹ Jun Liu³
Lingzi Hong² Caiwen Ding⁴ Jundong Li⁵ Xiaoming Zhai¹ Shaoyi Huang⁶
Jin Lu¹ Ninghao Liu¹ Wei Niu¹ Geng Yuan¹

¹University of Georgia ²University of North Texas ³Northeastern University
⁴University of Minnesota ⁵University of Virginia ⁶Stevens Institute of Technology

Abstract

Quantization reduces the deployment cost of large language models (LLMs). However, post-training quantization (PTQ) cannot fine-tune model parameters and often degrades accuracy in low-bit settings, while quantization-aware training (QAT) requires backpropagation and incurs high memory overhead. We propose ZeroQAT, a zeroth-order QAT framework supporting both weight and activation quantization. By using forward-only gradient estimation, ZeroQAT removes backpropagation and substantially reduces memory and computation while retaining end-to-end optimization benefits. We further introduce a lightweight variant of ZeroQAT for quantized fine-tuning, which freezes and pre-quantizes most parameters to further cut memory usage. Experiments show that ZeroQAT consistently outperforms representative PTQ and QAT baselines while requiring significantly less memory. For example, ZeroQAT enables fine-tuning of a 13B model at extremely low bit-widths (e.g., 2-4 bits) on a single 8GB GPU, and even allows fine-tuning a 6.7B model on a OnePlus 12 smartphone, demonstrating its practicality for end-to-end QAT on resource-limited edge devices.

1 Introduction

Large language models (LLMs) have emerged as essential tools for advancing natural language understanding and generation, driving progress in both research and industrial applications (Yang et al., 2019; Liu et al., 2019; Talmor et al., 2018; Chowdhery et al., 2023; Zheng et al., 2020). Despite their transformative potential, training and deploying these models incur extremely high computational and memory costs. Such requirements not only constrain accessibility and scalability but also limit practicality in resource-constrained environments, including mobile and edge devices, embedded systems, and even enterprise servers with strict hardware or budget limitations (Zeng et al.,

2024; Chen et al., 2024; Tan et al., 2025).

To address these challenges, model compression has been widely studied, with quantization emerging as one of the most effective techniques for deployment. Existing methods are mainly divided into post-training quantization (PTQ) and quantization-aware training (QAT). PTQ is simple and practical since it avoids retraining, whereas QAT typically achieves higher accuracy when sufficient resources are available. However, for LLMs, the memory cost of QAT is often prohibitive (Team et al., 2025), making PTQ the dominant choice in practice due to feasibility rather than superiority.

In low-bit settings, the key challenge is adapting to the distribution shifts introduced by quantization. Range-based PTQ methods (Jacob et al., 2018; Nagel et al., 2019; Xiao et al., 2023) provide limited adaptation and often suffer from noticeable accuracy loss. More advanced approximation-based PTQ methods (Nagel et al., 2020; Li et al., 2021; Frantar et al., 2022; Shao et al., 2023) better match full-precision outputs, but they still rely on indirect block-wise reconstruction objectives rather than true end-to-end optimization. This often leads to cumulative errors and objective mismatch, especially in low-bit regimes. The problem is even more severe for fine-tuned models, which are highly task-specific and sensitive to quantization perturbations (Dong et al., 2021).

QAT offers a more principled solution by incorporating quantization effects during training, enabling the model to adapt to quantization errors. Yet its reliance on backpropagation makes it difficult to apply to large models. Recent zeroth-order (ZO) optimization methods estimate gradients using only forward passes, avoiding the storage of activations and optimizer states and thus greatly reducing memory usage. This raises a natural question: *Can ZO enable memory-efficient QAT for high-quality low-bit quantization of LLMs?*

In this work, we propose ZeroQAT, the first end-

Table 1: Comparison of our method with existing methods. PEFT indicates parameter-efficient fine-tuning. WO and WA indicate weight-only and weight-activation quantization, respectively.

Method	Category	Quant Support	Low-bit performance		Memory Efficiency
			Pre-train	Fine-tune	
SmoothQuant	Range PTQ	WA	✗	✗	High
GPTQ	Approx PTQ	WO	✗	✗	High
OmniQuant	Approx PTQ	WA	✓	✗	Moderate
LLM-QAT	Full QAT	WA	✓	✓	Low
QLoRA	PEFT QAT	WO	✓	✓	Moderate
EfficientQAT	PEFT QAT	WO	✓	✓	High
ZeroQAT	Full/PEFT QAT	WA	✓	✓	High

to-end QAT framework that supports on-device quantization of both low-bit weights and activations. Here, end-to-end carries two meanings. First, the entire training and inference pipeline can be executed directly on-device, without offloading any computation to external servers. This enables users to adapt models locally under strict resource constraints. Second, end-to-end means that all trainable parameters are jointly optimized using the downstream task loss, rather than being updated block by block with auxiliary reconstruction objectives.

As shown in Table 1, ZeroQAT reduces the resource burden of conventional QAT while mitigating the accuracy loss commonly seen in PTQ. Unlike prior methods that require massive computing resources (Liu et al., 2023), ZeroQAT updates model parameters using gradients estimated purely from forward passes, reducing memory usage to inference-level and making QAT feasible even on edge devices. It further integrates learnable weight clipping and activation transformations, optimized jointly with model parameters via ZO. Moreover, a lightweight variant is devised for further memory reduction. Experiments on both quantized pre-training and fine-tuning show that ZeroQAT consistently outperforms representative PTQ and QAT baselines. For instance, it improves accuracy by 5.1% on average over five zero-shot tasks and 9.1% on four downstream tasks under 2-bit weight-only quantization. More importantly, ZeroQAT overcomes the memory barrier of QAT, enabling training of a 13B LLM on a single 8GB low-end GPU and even fine-tuning a 6.7B model on a OnePlus 12 smartphone. This capability makes end-to-end on-device QAT practical on resource-constrained edge devices.

In summary, our major contributions are as follows: 1) We conduct a preliminary study of PTQ and QAT in low-bit pre-training and fine-tuning,

revealing their weaknesses and causes of performance degradation. 2) We propose ZeroQAT, a novel end-to-end zeroth-order QAT framework that achieves high-quality low-bit quantization with inference-level cost. 3) We conduct extensive evaluation across LLM architectures, datasets, and quantization settings, showing consistent accuracy and memory improvements over prior PTQ and QAT baselines. 4) We validate on mobile devices, where ZeroQAT can fine-tune OPT-6.7B on the OnePlus 12 smartphone while full-precision ZO fine-tuning is infeasible, demonstrating its practicality for real-world deployment.

2 Background and Related Works

Quantization. In this work, we mainly study the widely used uniform quantization (Jacob et al., 2018) for its better efficiency. The quantization process can be formulated as:

$$\bar{\mathbf{X}}_{\text{INT}} = \text{clamp}\left(\left\lceil \frac{\mathbf{X}_{\text{FP16}}}{\Delta} \right\rceil + z, Q_N, Q_P\right) \quad (1)$$

where $\mathbf{X}$ is the floating-point tensor, $\bar{\mathbf{X}}$ is the quantized counterpart, $\lceil \cdot \rceil$ is rounding operation, N is the target bit number, Δ and z denote the step size and zero-point offset value respectively. For symmetric quantization, $Q_N = -2^{N-1}$, $Q_P = 2^{N-1} - 1$, $\Delta = \frac{\max(|\mathbf{X}|)}{Q_P}$ and $z = 0$. Whereas for asymmetric quantization, $Q_N = 0$, $Q_P = 2^N - 1$, $\Delta = \frac{\max(|\mathbf{X}|) - \min(|\mathbf{X}|)}{Q_P}$ and $z = -\lceil \frac{\min(|\mathbf{X}|)}{\Delta} \rceil$. In this paper, we focus on the asymmetric quantization scheme for its better accuracy.

Block-wise calibration. Block-wise calibration strategy is the most widely adopted approach in approximation-based PTQ, because it is relatively efficient in terms of memory, computation, and data usage. The key idea is to minimize quantization error via reconstruction objectives. For example, the widely used block-wise reconstruction loss mini-

Table 2: Results of applying different quantization methods on LLama-2-7B. [‡] indicates that the method is intrinsically not suitable for the setting; we report these results to illustrate its limitations.

Method	Category	Quantized Pre-training (PPL ↓)			Quantized Fine-tuning (Acc ↑)		
		W6A6	W2A16	W4A4	W6A6	W2A16g128	W4A4
ZO (FP16)	-		5.47			66.0	
Zero-shot	-		-			41.3	
SmoothQuant	Range-based PTQ	6.20	100.23 [‡]	83.12	57.2	27.7 [‡]	32.9 [‡]
OmniQuant	Approx-based PTQ	5.87	37.37	14.26	63.9	40.6 [‡]	38.8 [‡]
EfficientQAT	QAT	5.60	33.40	76.32 [‡]	66.4	45.4	28.6 [‡]
ZeroQAT	QAT	5.76	29.61	12.95	65.3	54.1	55.7

mizes the squared error, relative to the full precision layer output (Li et al., 2021; Shao et al., 2023). Formally, when both weights and activations are quantized, this can be stated as

$$\arg \min_{\bar{W}^l} \|W^l X^l - \bar{W}^l \bar{X}^l\|_2^2. \quad (2)$$

where $\bar{W}, \bar{X}$ are the quantized version of weight and activations, l indicates the l -th layer.

We present our related works section in Appendix A.

3 Challenge of Existing Methods

3.1 Challenge of Existing PTQ Methods

Range-based PTQ. These methods rescale or clip the ranges of weight and activation to reduce quantization error. They are computationally efficient and perform reasonably well at moderate bit-width. For example, SmoothQuant (Xiao et al., 2023) achieves a perplexity of 6.20 in W6A6 (i.e., quantization using 6-bit weight and 6-bit activation), close to the full-precision 5.47 (Table 2). However, their limited adaptation to distributional and semantic characteristics leads to severe degradation at low bit-widths. For example, under W4A4, SmoothQuant’s perplexity deteriorates to 83.12 versus 5.47 in full precision.

Approximation-based PTQ. These methods narrow the gap between quantized and full-precision outputs via techniques such as learned rounding or reconstruction, adapting to data distributions and model behavior. However, two issues remain and are exacerbated in low-bit quantization settings.

Here, we take a representative approximation-based PTQ method, OmniQuant (Shao et al., 2023), as an example to illustrate the two issues. 1) Cumulative error propagation. To measure error propagation, we compute relative loss reduction across layers, $\Delta_{Loss} = (\mathcal{L}_{before} -$

$\mathcal{L}_{after})/\mathcal{L}_{before}$, where $\mathcal{L}_{before}$ and $\mathcal{L}_{after}$ denote reconstruction loss before and after optimization. As shown in Figure 1, OmniQuant improves shallow layers, but benefits diminish in deeper ones, since each layer is optimized on activations already perturbed by prior quantization noise, making it increasingly difficult to suppress the reconstruction error. This cumulative error propagation constrains the overall quantization quality. 2) Objective inconsistency. OmniQuant uses block-wise reconstruction loss (see Eq.2) as training objective, assuming lower reconstruction loss is aligned with lower perplexity and better downstream accuracy. However, as shown in Figure 2, this alignment does not always hold, in several training stages (highlighted in red), reconstruction loss decreases while perplexity fluctuates. This indicates that local layer-level improvements do not reliably translate into global task-level gains, making reconstruction loss a suboptimal proxy for end-to-end performance, especially under low-bit quantization.

Failure on fine-tuned model. When PTQ is applied to fine-tuned LLMs, it often fails to preserve task accuracy under low-bit settings. As shown in Table 2, SmoothQuant maintains moderate accuracy at W6A6 (57.2% vs. 66.0% in FP16) but drops to 32.9% at W4A4. Similarly, OmniQuant achieves 63.9% at W6A6, close to FP16, yet falls to 38.8% at W4A4 despite optimization-based techniques. These results indicate that while PTQ remains viable at moderate bit-widths, its effectiveness collapses under aggressive compression, in some cases nearly destroying task performance.

3.2 Challenge of Existing QAT Methods

Compared with PTQ, QAT offers stronger adaptation by compensating for quantization errors during training. However, its computational and memory costs are prohibitive for LLMs (Liu et al.,

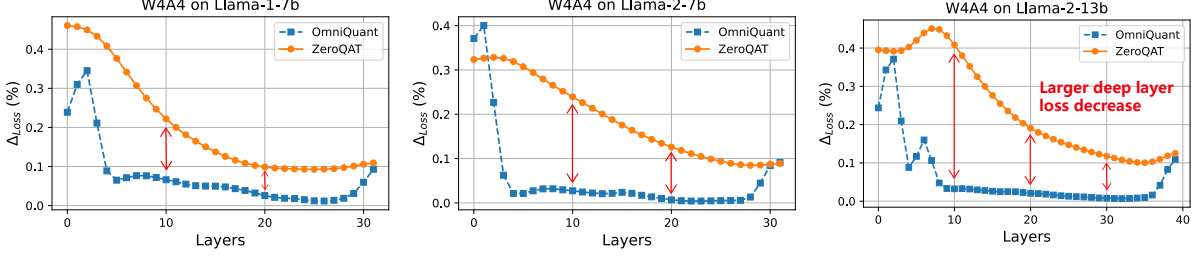


Figure 1: Comparison of the block-wise reconstruction loss reduction between OmniQuant (Shao et al., 2023) and our method. The X-axis is the index of layer, Y-axis measures the ratio of loss decrease.

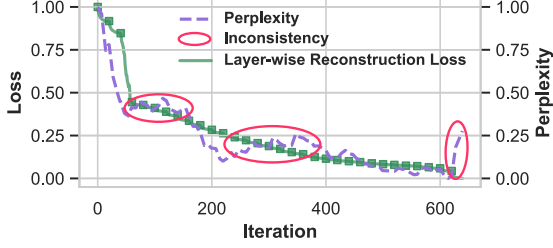


Figure 2: Objective inconsistency between the reconstruction loss used by approximation-based PTQ and final evaluation metrics.

2023). To reduce this overhead, later works combine QAT with parameter-efficient methods such as LoRA (Dettmers et al., 2023; Xu et al., 2023; Li et al., 2023) or update only quantizer parameters (Chen et al., 2025), achieving competitive results in weight-only quantization. Yet their effectiveness drops in low-bit joint weight-activation settings, as shown in Table 2. EfficientQAT maintains reasonable perplexity at W6A6 (5.60) and W2A16 (33.40), but degrades sharply at W4A4 (76.32), highlighting the difficulty of modeling dynamic activations.

Overall, although QAT methods can surpass PTQ in some settings, they have not consistently delivered strong results for both weight and activation quantization at aggressive bit-widths under realistic resource constraints. Motivated by this gap, we develop a ZO-based QAT framework that, to the best of our knowledge, is the first to maintain superior accuracy in both low-bit weight and activation settings.

4 ZeroQAT

4.1 Quantization-aware Zeroth-order Optimization

Unlike conventional first-order optimization that computes gradients via backpropagation, zeroth-order (ZO) optimization estimates them using only function queries through finite differences (Chen

et al., 2023; Liu et al., 2018; Ye et al., 2018). This avoids storing activations, backward gradients, and optimizer states, greatly reducing memory costs in LLM fine-tuning. For each random direction, ZO requires only two forward passes to approximate the gradient:

$$\hat{\nabla} \mathcal{L}(\bar{W}) = \frac{1}{q} \sum_{i=1}^q \left[\frac{\mathcal{L}(Q(W + \epsilon u_i)) - \mathcal{L}(Q(W - \epsilon u_i))}{2\epsilon} u_i \right] \quad (3)$$

where Q is the quantizer, $\bar{W}$ is the quantized parameters, $u_i \in \mathcal{N}(0, \mathbf{I})$ is a random perturbation, q is the number of directions, and $\epsilon > 0$ is a small scalar.

Following QAT practice, we maintain full-precision weights while using their quantized counterparts in forward passes. Unlike FO-QAT, ZeroQAT does not require the straight-through estimator (STE) (Bengio et al., 2013), since gradients are estimated directly via zeroth-order finite differences, bypassing the non-differentiability of the quantizer. Given a learning rate η and a mini-batch $\mathcal{B}_t$ at iteration t , the update rule becomes:

$$W_{t+1} = W_t - \eta \hat{\nabla} \mathcal{L}(\bar{W}_t; \mathcal{B}_t). \quad (4)$$

In ZeroQAT, the ZO estimator remains unbiased with respect to the gradient of a smoothed quantized objective, which ensures standard convergence guarantees. In contrast, QAT methods based on the STE rely on a hand-crafted surrogate gradient that introduces inherent bias. This bias becomes particularly severe in low-bit regimes, where the true smoothed gradients are already small but STE still produces large surrogate updates, leading to unstable or suboptimal convergence. A formal analysis and quantitative bounds on this bias are provided in Appendix G.

4.2 Adaptive Outlier Mitigation

Adaptive per-channel scaling. Activation outliers enlarge the dynamic range of specific channels and

degrade quantization accuracy. Prior work (Xiao et al., 2023; Wei et al., 2022; Shao et al., 2023; Sun et al., 2025) mitigates this issue by applying channel-wise scaling $\text{diag}(\mathbf{c}) \in \mathbb{R}^n$ to activations and weights in a mathematically equivalent form, where $\mathbf{c}$ is a statistical or learnable scaler, thereby transferring activation quantization difficulty to weights with more uniform distributions.

Adaptive weight & activation clipping. To further suppress residual outliers after per-channel smoothing, we introduce learnable clipping thresholds for both weights and activations. For a generic tensor $\mathbf{T} \in \mathbf{X}, \mathbf{W}$ in a linear layer, we apply asymmetric learnable clipping in a unified form, $\bar{\mathbf{T}} = \text{clamp}(\mathbf{T}, \alpha_{\mathbf{T}} \cdot Q, \beta_{\mathbf{T}} \cdot Q)$, where $\alpha_{\mathbf{T}}, \beta_{\mathbf{T}} \in (0, 1)$ is a layer-specific clipping threshold and jointly optimized with other parameters. This formulation enables end-to-end adaptation of clipping ranges for both weights and activations, yielding lower quantization error without additional inference overhead. Intuitively, for weight or activation distributions with near-uniform distributions after smoothing, α and β converge to similar values, resulting in a tight clipping range that preserves precision. In contrast, for biased distributions, α and β adapt to asymmetrically clip the dynamic range, thereby mitigating the impact of outliers.

Adaptive transformation. Following prior work (Ashkboos et al., 2024; Liu et al., 2025), we apply linear transformations to smooth activations and suppress outliers before quantization, thereby improving the flatness of both weight and activation distributions. Concretely, given a linear layer $\mathbf{Y} = \mathbf{XW}$, the transformation can be written in a mathematically equivalent form as $\mathbf{Y} = (\mathbf{XP}^{-1})(\mathbf{PW})$, where the transformation matrix $\mathbf{P}$ redistributes activation magnitudes and mitigates outliers. For efficient implementation, we are inspired by FlatQuant (Sun et al., 2025) and factorize large affine transformations via matrix decomposition, which significantly reduces memory consumption and computational overhead.

In practice, we design six transformations $\mathcal{P} = \{P_a, P_o, P_k, P_v, P_f, P_d\}$, which are applied to the QKV projections, output projection, key cache, value cache, feed-forward layer, and down-projection layer, respectively. To minimize runtime overhead, different transformations are selectively fused or merged into the corresponding weights, ensuring that the overall inference speed is not noticeably affected.

For all the quantization parameters, previous

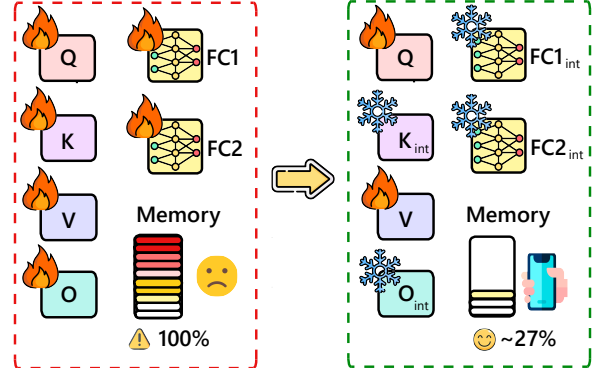


Figure 3: lightweight ZeroQAT for fine-tuning.

works generally use hand-crafted or block-wise calibrated scaling, which lacks end-to-end optimization and often leads to suboptimal performance. In contrast, our QAT framework jointly optimizes these parameters along with model weights, enabling adaptive suppression of outliers during training.

4.3 Lightweight ZeroQAT for Memory Reduction in Quantized Fine-tuning

We further propose a lightweight variant of ZeroQAT designed specifically for quantized fine-tuning to substantially reduce the fine-tuning memory footprint. Unlike backpropagation-based methods, where memory is dominated by weights, activations, and optimizer states, ZeroQAT’s cost mainly comes from the parameters actively updated during fine-tuning. Pre-quantizing the entire model could further reduce memory, but this fails in practice. As small ZO perturbations are rounded away while large ones destabilize training, making naive full-model pre-quantization unsuitable.

To overcome this, we introduce a lightweight variant. Most parameters are frozen and pre-quantized, while only the query (Q) and value (V) matrices of attention blocks are kept in full precision, as illustrated in Figure 3. Thus, memory use comes from the full-precision Q and V plus quantized frozen weights. This design substantially reduces the fine-tuning footprint while retaining sufficient trainable capacity for adaptation. Although pre-quantizing frozen weights brings only limited memory savings in FO due to the dominant memory cost coming from activation and optimizer, it becomes particularly beneficial in the ZO setting. As model weights account for a much larger fraction of the overall memory footprint, pre-quantizing frozen weights becomes particularly beneficial in the ZO setting. Therefore, it enables

Table 3: Weight-only and weight-activation quantization results of LLama-series models on two datasets: WikiText2 (WIKI), and C4.

LLama / PPL ↓		LLama1-7B		LLama1-13B		LLama2-7B		LLama2-13B		LLama3-8B	
Task		WIKI	C4	WIKI	C4	WIKI	C4	WIKI	C4	WIKI	C4
FP16	-	5.68	7.08	5.09	6.61	5.47	6.97	4.88	6.46	6.14	9.45
W2A16	RTN	1.1e5	1.3e5	6.8e4	5.6e4	3.8e4	4.8e4	4.2e4	1.2e3	4.2e4	5.6e4
	GPTQ	5.6e4	689.13	5.5e3	6.97	7.7e3	NAN	2.1e3	323.12	4.2e3	1.2e3
	OmniQuant	15.47	24.89	13.21	18.31	37.37	90.64	17.21	26.76	28.48	56.64
	ZeroQAT	9.65	13.55	9.09	11.87	17.18	28.36	8.67	13.68	9.14	12.89
W4A4	SmoothQuant	25.25	32.32	40.05	47.18	83.12	77.27	35.88	43.19	210.19	187.93
	OmniQuant	11.26	14.51	10.87	13.78	14.26	18.02	12.30	14.55	13.41	16.83
	SpainQuant	6.44	8.13	5.91	7.79	6.14	9.19	5.44	7.48	7.96	13.45
	FlatQuant	6.33	7.79	5.47	7.37	5.79	7.79	5.12	7.11	6.98	11.13
	ZeroQAT	6.17	8.08	5.66	7.56	5.78	7.93	5.27	7.43	6.12	10.43

Table 4: Weight-only and weight-activation quantization results of LLama models. This table reports the accuracy of 5 zero-shot tasks. Refer to Table E.1 for more results.

LLama / Acc ↑	Method	PIQA	ARC-e	ARC-c	HellaSwag	Winogrande	Avg.
LLama-1-7B	FP16	77.47	72.38	41.46	73.00	67.07	65.26
	SpainQuant	73.81	66.58	34.16	67.02	59.95	60.30
	FlatQuant	75.73	68.87	36.88	69.31	65.46	63.17
	ZeroQAT	76.98	69.32	36.19	70.86	64.87	63.64
LLama-2-7B	FP16	79.05	74.54	46.16	75.98	69.06	68.96
	SpainQuant	75.30	65.32	39.42	71.45	63.46	62.99
	FlatQuant	77.26	72.05	43.26	73.64	69.53	67.14
	ZeroQAT	76.86	68.67	42.39	74.43	68.35	66.14
LLama-3-8B	FP16	80.74	77.57	53.50	79.12	72.93	72.77
	SpainQuant	76.66	71.38	45.73	74.07	66.38	66.84
	FlatQuant	79.16	75.80	50.00	76.80	72.69	69.89
	ZeroQAT	79.48	77.24	51.08	77.06	71.55	71.28

fine-tuning large models such as OPT-13B under low-bit settings with memory as low as 6.8 GB (in Table 7), far lighter than existing QAT baselines.

5 Experiment

We evaluate ZeroQAT on quantized pre-training and fine-tuning, with ablations and efficiency analysis of memory and speed. Experiments are conducted on an NVIDIA A100 GPU and a OnePlus 12 device (Snapdragon 8 Gen 3). Hyperparameter settings are provided in Appendix B.1, and all results are averaged over three runs.

5.1 ZeroQAT for Quantized Pre-training

Training and evaluation. For the parameters of smoothing (channel-wise scaling and transformation) and clipping, we leverage reconstruction loss for a lightweight initialization, and then jointly train with the model via ZO. For LLama-series weight-only quantization, we retain only clipping. Pre-training uses mixed segments from WikiText2 and C4, with perplexity measured on the pre-training context datasets. We further evaluate zero-

shot accuracy on five datasets under GPTQ (Frantar et al., 2022) settings with lm-eval-harness (Gao et al., 2024). More experimental and baseline details are provided in Appendix B.2.

Perplexity results. We evaluate the intrinsic language modeling performance of quantized models using perplexity, with results for the LLaMA and OPT families reported in Tables 3 and D.2. Under the relatively mild W6A6 setting, all methods achieve near lossless performance compared to full precision. In contrast, under more aggressive settings (W2A16 and W4A4), ZeroQAT consistently outperforms baseline methods by enabling stronger adaptation through end-to-end training, resulting in lower perplexity across model families and datasets. These results demonstrate the effectiveness of ZeroQAT in preserving language modeling performance under severe quantization.

Zero-shot accuracy results. Table 4 reports the zero-shot results of the LLama-series models on five reasoning datasets evaluated by accuracy. As expected, the FP16 setting achieves the highest average accuracy, serving as the upper bound. With

Table 5: Weight-only and weight-activation quantized fine-tuning results of OPT models. The table reports the accuracy or F1 score of the corresponding tasks. g128 indicates the group size is 128.

OPT / Acc $\uparrow$		OPT-2.7B				OPT-6.7B				OPT-13B			
Task		SST-2	CB	SQuAD	DROP	SST-2	CB	SQuAD	DROP	SST-2	CB	SQuAD	DROP
Zero-shot		56.3	50.0	29.8	10.0	64.2	50.0	37.9	13.1	58.8	46.4	46.2	14.6
FP16 (ZO)	-	90.0	69.6	68.7	22.9	90.2	71.4	76.0	26.4	91.4	67.9	84.7	30.9
W2A16g128	RTN	44.4	44.6	0.0	0.0	59.2	50.0	0.0	0.0	53.5	50.0	0.0	0.0
	QuZO	47.2	46.3	0.0	0.0	60.6	55.3	0.0	0.0	59.6	51.8	0.0	0.0
	QLoRA	61.2	51.8	0.0	8.2	64.8	58.9	0.0	0.0	63.8	69.6	0.0	0.0
	OmniQuant	72.8	55.4	16.5	4.4	61.6	55.3	27.7	12.6	62.6	29.8	38.8	16.4
	EfficientQAT	76.6	57.1	29.0	12.6	75.6	58.9	32.4	14.6	81.2	62.5	46.7	16.9
	ZeroQAT	87.8	64.8	52.8	16.2	88.6	67.4	61.3	22.4	89.4	63.8	68.8	25.9
W4A4	QuZO	53.4	53.6	0.0	0.0	60.3	57.3	0.0	0.0	57.4	51.8	0.0	0.0
	SmoothQuant	56.0	55.4	7.6	5.4	58.8	50.0	12.8	6.2	57.5	52.4	13.4	7.1
	OmniQuant	59.2	60.7	22.1	6.7	61.2	48.2	24.7	11.7	59.2	50.0	28.8	13.5
	FlatQuant	79.7	62.5	38.7	12.4	82.3	57.1	47.6	17.9	85.7	60.7	58.9	22.8
	ZeroQAT	88.4	67.8	59.7	18.3	89.3	64.3	64.1	21.3	89.8	66.1	72.4	27.3

the 4-bit weight and activation quantization, ZeroQAT outperforms other quantization approaches, yielding higher average accuracy across models. This result demonstrates that ZeroQAT maintains strong task generalization even when quantization is pushed to low-bit precision.

It is worth noting that although strong baseline methods such as FlatQuant can achieve performance comparable to ZeroQAT in the quantized pre-training setting, the advantages of our method become more evident in the subsequent quantized fine-tuning experiments and efficiency analysis.

5.2 ZeroQAT for Quantized Fine-tuning

Training and evaluation. Following prior work (Pan et al., 2024; Chen et al., 2025), we fine-tune models on a small subset of Alpaca (Taori et al., 2023) and evaluate the reasoning performance. Moreover, we adopt a few-shot fine-tuning protocol for evaluation following previous work (Malladi et al., 2023). For evaluation on PTQ methods, we first fine-tune the models in full precision using ZO, and then apply the corresponding quantization method. In contrast, QAT methods, like EfficientQAT and our ZeroQAT, directly produce quantized models during fine-tuning without the need for a separate PTQ stage. Additional experimental and baseline details are provided in Appendix B.3.

Results. The results are summarized in Table 5. Fine-tuning adapts model parameters to narrow task-specific optima (Dong et al., 2021), which increases their sensitivity to quantization noise. Consequently, less adaptive PTQ methods suffer from severe degradation in low-bit settings. By comparison,

ZeroQAT consistently delivers higher accuracy across all tasks and model scales, in some cases approaching FP16 performance. For example, under the W4A4 setting, ZeroQAT achieves about 89% accuracy on SST-2 across the three OPT models, whereas the strongest baseline methods achieve less than 80%. We also conduct fine-tuning on Alpaca, with results shown in Table 6. ZeroQAT again outperforms prior methods across different bit-widths and model sizes.

Table 6: Averaged accuracy over 5 zero-shot datasets after fine-tuning. Results on MMLU and GSM8K are presented in Table E.2 and Table E.3.

Method	#Bits	7B	13B
-	FP	67.0	69.3
QLoRA w/GPTQ	W2A16	31.8	32.4
QA-LoRA	W2A16	34.6	37.3
IR-QLoRA	W2A16	34.4	36.3
PEQA	W2A16	35.2	34.8
EfficientQAT	W2A16	49.1	52.1
ZeroQAT	W2A16	59.4	62.7
SmoothQuant	W4A4	37.4	41.6
OmniQuant	W4A4	52.3	54.2
FlatQuant	W4A4	58.8	60.7
ZeroQAT	W4A4	63.1	66.8

5.3 Efficiency of ZeroQAT

To highlight the advantage that our method enables generating a quantized and fine-tuned model through a lightweight end-to-end pipeline, we evaluate the efficiency of ZeroQAT on both a GPU server and a mobile device to demonstrate its practicality across different platforms. Implementing and optimizing such a pipeline on mobile devices is non-trivial due to their limited computational

Table 7: Memory consumption and wallclock time per update during quantized pre-training and fine-tuning under the W2A16g128 setting. Since ZeroQAT only stores the weights in memory, its memory usage remains unaffected by batch size. Four and sixteen is the ideal batch size in FlatQuant and EfficientQAT, respectively.

Method	OPT-1.3B		OPT-2.7B		OPT-6.7B		OPT-13B	
	Memory	Time	Memory	Time	Memory	Time	Memory	Time
Quantized Pre-training (avg sequence length = 2048)								
LLM-QAT (bsz=1)	28.8GB	1.00s	58.6GB	1.64s	~166GB	~5.0s	~337GB	~15.5s
FlatQuant (bsz=1)	7.2GB	1.12s	8.5GB	1.83s	13.9GB	3.04s	20.1GB	5.14s
ZeroQAT (bsz=1)	3.1GB	0.58s	6.1GB	0.98s	14.2GB	1.77s	26.6GB	3.12s
FlatQuant (bsz=4)	17.4GB	2.55s	21.5GB	4.03s	27.2GB	6.35s	33.8GB	11.81s
ZeroQAT (bsz=4)	3.1GB	1.72s	6.1GB	2.76s	14.2GB	4.48s	26.6GB	7.74s
Quantized Fine-tuning (max sequence length = 384)								
EfficientQAT (bsz=1)	2.1GB	0.13s	3.1GB	0.21s	4.4GB	0.36s	7.3GB	0.67s
ZeroQAT (bsz=1)	0.8GB	0.04s	1.5GB	0.07s	3.7GB	0.18s	6.8GB	0.32s
EfficientQAT (bsz=16)	5.9GB	0.69s	8.1GB	1.10s	11.9GB	1.70s	17.2GB	3.26s
ZeroQAT (bsz=16)	0.8GB	0.31s	1.5GB	0.53s	3.7GB	0.94s	6.8GB	1.73s

Table 8: Evaluation of memory consumption and speed on a OnePlus 12 smartphone under W4A4 quantization. Prompts of 384 tokens are used in inference, and OOM indicates out of memory.

Stage	Metrics	OPT-1.3B		OPT-2.7B		OPT-6.7B	
		FP16	ZeroQAT	FP16	ZeroQAT	FP16	ZeroQAT
Fine-tuning	Latency	11.2s	7.8s	19.6s	12.3s	/	29.1s
	Weight memory	2.6GB	0.9GB	5.4GB	1.8GB	13.4GB	4.6GB
	Running memory	3.5GB	1.2GB	8.1GB	2.6GB	OOM	6.4GB
Inference	Token / s	10.9	15.4	7.58	11.0	3.13	4.76
	Speed up	1.0×	1.41×	1.0×	1.45×	1.0×	1.52×

and memory resources. Therefore, we provide additional details on the implementation and optimization of our method on mobile devices in Appendix F.

Server-side efficiency. Table 7 compares memory requirements and wallclock time per update between our method and baselines. For quantized pre-training, ZeroQAT reduces memory usage by 89-92% relative to the costly LLM-QAT, while also accelerating training. Compared to the PTQ method FlatQuant, ZeroQAT offers clear advantages, for instance, it halves memory use (OPT-1.3B: 6.1GB to 3.1GB) and achieves about 1.5× faster updates (OPT-2.7B: 1.49s to 0.98s). For quantized fine-tuning, ZeroQAT’s memory-efficient design requires storing only weights of query and value heads in attention blocks, making memory usage independent of batch size. Against EfficientQAT, it consistently saves memory and improves throughput, especially on smaller models such as OPT-1.3B, reducing memory by 86% (5.9GB to 0.8GB) and wallclock time by 55% (0.69s to 0.31s) with the same batch size.

On-device efficiency. Table 8 compares FP16 baseline with ZeroQAT under W4A4 for OPT-1.3B, 2.7B, and 6.7B models. The results were collected

on a OnePlus 12 smartphone with a Snapdragon 8 Gen 3 SoC and 16GB RAM. ZeroQAT reduces fine-tuning latency by 30% and 37% for OPT-1.3B and OPT-2.7B, respectively, while cutting running memory from 3.5GB to 1.2GB and from 8.1GB to 2.6GB. For OPT-6.7B, FP16 fine-tuning is infeasible (OOM), whereas ZeroQAT runs within 6.4GB memory with 29.1s latency. During inference, ZeroQAT further achieves $1.41\times$ - $1.52\times$ higher token throughput, demonstrating its practicality on resource-constrained devices.

6 Conclusion

In this paper, we propose ZeroQAT, a zeroth-order QAT framework that supports both weight and activation quantization at extremely low-bit setting. We introduce adaptive smoothing and an adaptive weight quantizer to reduce quantization error, along with a lightweight variant that freezes and quantizes part of the model to lower fine-tuning memory cost. Experiments on quantized pre-training, fine-tuning, and on-device deployment show that ZeroQAT consistently outperforms PTQ and QAT baselines in both accuracy and efficiency, enabling fine-tuning of large LLMs on a OnePlus 12 smartphone under strict memory constraints.

7 Limitations

While ZeroQAT demonstrates strong performance across multiple model scales and deployment settings, our current evaluation is limited to LLMs up to 13B parameters, as our method primarily targets resource-limited platforms (e.g., mobile devices). Models larger than this scale (e.g., 30B parameters and beyond) typically fall outside the resource constraints of such platforms and are therefore less aligned with our primary deployment scenario. Nevertheless, extending the proposed framework to larger models remains an interesting direction for future work.

In addition, the requirement for more training iterations is an inherent limitation of our approach, as well as of ZO-based methods in general. Developing more efficient zero-order gradient estimators could substantially reduce the number of required optimization steps. Given that our focus is on memory efficiency rather than speed, and acceleration techniques are orthogonal to and fully compatible with our framework. Exploring these improvements represents an important, but separate, direction for future work.

8 Claim of LLM Usage

In this work, large language models (LLMs) were used solely as a general-purpose writing assistant. Their role was limited to correcting grammar, fixing typographical errors, and polishing the language for clarity and readability.

Acknowledgment

This work is partly supported by the National Science Foundation IIS-2223768 and DRL-2507128.

References

- Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoeffler, and James Hensman. 2024. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240.
- Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.
- Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.
- Aochuan Chen, Yimeng Zhang, Jinghan Jia, James Diefenderfer, Jiancheng Liu, Konstantinos Parasyris, Yihua Zhang, Zheng Zhang, Bhavya Kailkhura, and Sijia Liu. 2023. Deepzero: Scaling up zeroth-order optimization for deep model training. *arXiv preprint arXiv:2310.02025*.
- Hongzheng Chen, Jiahao Zhang, Yixiao Du, Shaojie Xiang, Zichao Yue, Niansong Zhang, Yaohui Cai, and Zhiru Zhang. 2024. Understanding the potential of fpga-based spatial acceleration for large language model inference. *ACM Transactions on Reconfigurable Technology and Systems*.
- Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. 2025. Efficientqat: Efficient quantization-aware training for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.
- Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. 2017. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 15–26.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.
- Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.
- Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115.
- Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. 2018. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. *Advances in neural information processing systems*, 31.

- Amit Dhurandhar, Tejaswini Pedapati, Avinash Balakrishnan, Pin-Yu Chen, Karthikeyan Shanmugam, and Ruchir Puri. 2019. Model agnostic contrastive explanations for structured data. *arXiv preprint arXiv:1906.00117*.
- Xinshuai Dong, Anh Tuan Luu, Min Lin, Shuicheng Yan, and Hanwang Zhang. 2021. How should pre-trained language models be fine-tuned towards adversarial robustness? *Advances in Neural Information Processing Systems*, 34:4356–4369.
- Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*.
- Elias Frantar, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.
- Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac’h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. [The language model evaluation harness](#).
- Jiaqi Gu, Chenghao Feng, Zheng Zhao, Zhoufeng Ying, Ray T Chen, and David Z Pan. 2021. Efficient on-chip learning for optical neural networks through power-aware sparse zeroth-order optimization. In *Proceedings of the AAAI conference on artificial intelligence*, pages 7583–7591.
- Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713.
- Jeonghoon Kim, Jung Hyun Lee, Sungdong Kim, Joon-suk Park, Kang Min Yoo, Se Jung Kwon, and Dongsoo Lee. 2023. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. *Advances in Neural Information Processing Systems*, 36:36187–36207.
- Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. 2023. Loftq: Lora-fine-tuning-aware quantization for large language models. *arXiv preprint arXiv:2310.08659*.
- 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. *arXiv preprint arXiv:2102.05426*.
- Sijia Liu, Bhavya Kailkhura, Pin-Yu Chen, Paishun Ting, Shiyu Chang, and Lisa Amini. 2018. Zeroth-order stochastic variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [Roberta: A robustly optimized BERT pretraining approach](#). *CoRR*, abs/1907.11692.
- Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*.
- Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Teian, and Tijmen Blankevoort. 2025. Spinqant: Llm quantization with learned rotations. In *The Thirteenth International Conference on Learning Representations*.
- Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. 2023. Fine-tuning language models with just forward passes. *Advances in Neural Information Processing Systems*, 36:53038–53075.
- Mitch Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.
- Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. 2020. Up or down? adaptive rounding for post-training quantization. In *International conference on machine learning*, pages 7197–7206. PMLR.
- Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. 2019. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1325–1334.
- Wei Niu, Jiexiong Guan, Yanzhi Wang, Gagan Agrawal, and Bin Ren. 2021. Dnnfusion: accelerating deep neural networks execution with advanced operator fusion. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 883–898.
- Wei Niu, Md Musfiqur Rahman Sanim, Zhihao Shu, Jiexiong Guan, Xipeng Shen, Miao Yin, Gagan Agrawal, and Bin Ren. 2024. Smartmem: Layout

- transformation elimination and adaptation for efficient dnn execution on mobile. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, pages 916–931.
- Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. 2024. Lisa: Layer-wise importance sampling for memory-efficient large language model fine-tuning. *Advances in Neural Information Processing Systems*, 37:57018–57049.
- Haotong Qin, Xudong Ma, Xingyu Zheng, Xiaoyang Li, Yang Zhang, Shouda Liu, Jie Luo, Xianglong Liu, and Michele Magno. 2024. Accurate lora-finetuning quantization of llms via information retention. *arXiv preprint arXiv:2402.05445*.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.
- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.
- Sifeng Shang, Jiayi Zhou, Chenyu Lin, Minxian Li, and Kaiyang Zhou. 2025. Fine-tuning quantized neural networks with zeroth-order optimization. *arXiv preprint arXiv:2505.13430*.
- 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*.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.
- Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, and 1 others. 2025. Flatquant: Flatness matters for llm quantization. In *Forty-second International Conference on Machine Learning*.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.
- Qitao Tan, Sung-En Chang, Rui Xia, Huidong Ji, Chence Yang, Ci Zhang, Jun Liu, Zheng Zhan, Zhenman Fang, Zhou Zou, and 1 others. 2025. Perturbation-efficient zeroth-order optimization for hardware-friendly on-device training. *arXiv preprint arXiv:2504.20314*.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.
- Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, and 1 others. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.
- Anirudh Vemula, Wen Sun, and J Bagnell. 2019. Contrasting exploration in parameter and action space: A zeroth-order optimization perspective. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2926–2935. PMLR.
- Astha Verma, Siddhesh Bangar, A Venkata Subramanyam, Naman Lal, Rajiv Ratn Shah, and Shin’ichi Satoh. 2023. Certified zeroth-order black-box defense with robust unet denoiser. *arXiv preprint arXiv:2304.06430*.
- Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. 2022. Outlier suppression: Pushing the limit of low-bit transformer language models. *Advances in Neural Information Processing Systems*, 35:17402–17414.
- Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.
- Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, Xiaopeng Zhang, and Qi Tian. 2023. Qa-lora: Quantization-aware low-rank adaptation of large language models. *arXiv preprint arXiv:2309.14717*.
- Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*.
- Haishan Ye, Zhichao Huang, Cong Fang, Chris Junchi Li, and Tong Zhang. 2018. Hessian-aware zeroth-order optimization for black-box adversarial attack. *arXiv preprint arXiv:1812.11377*.
- Zhihang Yuan, Lin Niu, Jiawei Liu, Wenyu Liu, Xinggang Wang, Yuzhang Shang, Guangyu Sun, Qiang Wu, Jiaxiang Wu, and Bingzhe Wu. 2023. Rptq: Reorder-based post-training quantization for large language models. *arXiv preprint arXiv:2304.01089*.

- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.
- Shulin Zeng, Jun Liu, Guohao Dai, Xinhao Yang, Tianyu Fu, Hongyi Wang, Wenheng Ma, Hanbo Sun, Shiyao Li, Zixiao Huang, and 1 others. 2024. Flightllm: Efficient large language model inference with a complete mapping flow on fpgas. In *Proceedings of the 2024 ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 223–234.
- Minghang Zheng, Peng Gao, Renrui Zhang, Kunchang Li, Xiaogang Wang, Hongsheng Li, and Hao Dong. 2020. End-to-end object detection with adaptive clustering transformer. *arXiv preprint arXiv:2011.09315*.
- Jiajun Zhou, Yifan Yang, Kai Zhen, Ziyue Liu, Yequan Zhao, Ershad Banijamali, Athanasios Mouchtaris, Ngai Wong, and Zheng Zhang. 2025. Quzo: Quantized zeroth-order fine-tuning for large language models. *arXiv preprint arXiv:2502.12346*.
- Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*.

A Related work

A.1 Model Quantization

Quantization techniques aim to properly map the original continuous real values to a discrete low-bit format (e.g., INT8 or INT4), leading to significant memory saving and inference acceleration while maintaining the performance (Zhou et al., 2016). Quantization techniques can be generally divided into two categories: Post-training quantization (PTQ) and quantization-aware training (QAT). The QAT method generally yields better results due to better adaptation capability, but the high retraining cost (in both memory and computation) has discouraged many researchers. Therefore, most of the LLM quantization works focus on PTQ methods, which can be mainly divided into range-based PTQ (Jacob et al., 2018; Nagel et al., 2019; Xiao et al., 2023) and approximation-based PTQ methods (Nagel et al., 2020; Li et al., 2021; Frantar et al., 2022; Shao et al., 2023). The range-based PTQ typically relies on static analysis, where the range (e.g., minimum and maximum values) of weights or activations is collected to determine quantization parameters. The approximation-based PTQ methods, with more adaptation, explicitly frame quantization as an error minimization problem, optimizing quantized parameters to closely approximate the full-precision model outputs.

A.2 Zeroth-order Optimization

Zeroth-order optimization (ZO), which estimates gradients using only function evaluations, has emerged as an attractive alternative to classical first-order (FO) methods. Compared to FO approaches, ZO eliminates the need for backpropagation, thereby simplifying implementation and significantly reducing memory consumption. This makes it appealing in scenarios such as adversarial attack and defense (Chen et al., 2017; Ye et al., 2018; Verma et al., 2023), machine learning explainability (Dhurandhar et al., 2018, 2019), reinforcement learning (Vemula et al., 2019), and on-chip training (Gu et al., 2021). Despite these successes, ZO optimization has been primarily applied to relatively small-scale problems, since its convergence is generally slower and suffers from high variance due to random search. These challenges are exacerbated in large-scale settings such as LLM fine-tuning, where dimensionality and resource constraints amplify the difficulty.

Several recent works explore leveraging zeroth-

order optimization (ZO) for LLM quantization (Zhou et al., 2025; Shang et al., 2025). Our approach differs from them in three key aspects. First, our forward pass strictly follows the standard low-bit quantization used in conventional QAT, ensuring that weights remain true low-bit representations. In contrast, Zhou et al. (2025) uses different quantizers for perturbation and weight updates, resulting in forward-pass weights that are effectively floating-point values. Similarly, Shang et al. (2025) adopts different scales for quantization and dequantization, which also breaks the strict low-bit constraint. As a result, these methods cannot fully leverage integer-level multiplications in low-bit computation kernels. Second, with the proposed adaptive outlier smoothing and clipping, jointly optimized with weights during training, our method is the first ZO-based framework supporting simultaneous low-bit quantization of both weights and activations, while prior work only supports weight quantization at moderate bit widths. Third, existing ZO-based quantization studies do not compare against state-of-the-art LLM quantization methods. Consequently, it is difficult to assess how ZO-based approaches perform relative to established quantization techniques. By conducting comprehensive comparisons with representative PTQ and QAT methods, our work provides the first systematic evaluation of ZO-based quantization under standard LLM quantization benchmarks, serving as a reference point for future research that combines ZO optimization with quantization.

B Experimental Settings

Quantization settings. To comprehensively evaluate our method, we consider both weight-only and weight-activation quantization, as they represent distinct deployment scenarios. For weight-activation quantization, we adopt per-channel weight quantization and per-token activation quantization, following prior work (Dettmers et al., 2022; Shao et al., 2023). For weight-only quantization, we apply a group-wise strategy, where the weight matrix is partitioned into groups of a fixed size, and each group is assigned its own scale and zero point. Formally, for example, W2A16g128 refers to 2-bit weight-only quantization with 128 as the group size. When g is omitted (e.g., W2A16), the default group size is set to the number of channels, corresponding to per-channel quantization.

B.1 Hyperparameter Setting

We use the hyperparameters in Table B.1 for experiments on quantized pre-training and quantized fine-tuning. Specifically, pre-training prefers a smaller learning rate and a smaller perturbation for stable convergence, while for fine-tuning, we can use more aggressive optimization. Moreover, larger models prefer smaller learning rate and a smaller perturbation, while smaller models tend to have the opposite.

B.2 Settings of Quantized Pre-training

Training and evaluation. Zeroth-order optimization has been shown to benefit from strong initialization (Malladi et al., 2023). To provide a stable starting point, we adopt a lightweight initialization strategy based on channel-wise scaling and shifting. Specifically, we pre-train quantized models with OmniQuant (Shao et al., 2023) for a few epochs (2 epochs in the W4A4 setting and 4 epochs in the W2A16 setting), which corresponds to roughly 10% of the full OmniQuant training cost. This initialization enables ZO to more effectively refine the quantization scales and shift factors. But for the Llama-series weight-only quantization, we remove the smoothing and only maintain clipping, as smoothing only provides limited improvement. For quantized pre-training, we randomly select token segments with length 2048 and then calculate perplexity over WikiText2 (Merity et al., 2016), PTB (Marcus et al., 1994), and C4 (Raffel et al., 2020). To avoid overfitting on one specific dataset, half of the segment samples are from WikiText2 and half from C4, while the total data size is kept the same as in previous work (Shao et al., 2023; Dettmers et al., 2022) and set as 128. We further assess zero-shot accuracy on a range of tasks, including PIQA (Bisk et al., 2020), ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021). We adhere to the GPTQ (Frantar et al., 2022) settings for language generation experiments, and leverage the lm-eval-harness (Gao et al., 2024) tool for the evaluation of all zero-shot tasks.

Baselines. We mainly compared with PTQ methods. For weight-only quantization, we compare with the vanilla round-to-nearest (RTN), GPTQ (Frantar et al., 2022). For weight-activation quantization, we compare our method with SmoothQuant (Xiao et al., 2023), RPTQ (Yuan et al., 2023), OutlierSuppression+ (OS+) (Wei et al.,

2022), OmniQuant (Shao et al., 2023), and one QAT method LLM-QAT (Liu et al., 2023). We keep the quantization setting of SmoothQuant and Outlier Suppression+ with per-channel weight quantization and per-token activation quantization for fair comparisons. Moreover, we compare with two more advanced spin transformation-based quantization methods, Spinquant (Liu et al., 2025) and Flatquant (Sun et al., 2025).

For the quantized pre-training setting, we do not compare with backpropagation-based QAT methods due to feasibility constraints. Specifically, pre-training uses long sequences (e.g., 2048 tokens), and backpropagation-based QAT requires storing full activations, whose memory cost grows quadratically with sequence length. In practice, in the pre-training setting, training a 4-bit 7B model with QAT like QLoRA requires over 80 GB of GPU memory, which is infeasible under our target low-resource setting. In contrast, in QAT-feasible scenarios (i.e., quantized fine-tuning with shorter sequences), we do compare against multiple QAT baselines, shown in quantized fine-tuning experiments.

B.3 Settings of Quantized fine-tuning

Training and evaluation. Following existing works (Chen et al., 2025; Pan et al., 2024), we fine-tune models on a small subset of the Alpaca dataset (Taori et al., 2023), and report the average accuracy on datasets including MMLU, PIQA, ARC, HellaSwag, and Winogrande. Moreover, we fine-tune and evaluate on two classification datasets, SST-2 (Socher et al., 2013) and CB (De Marneffe et al., 2019), and two question answering datasets, SQuAD (Rajpurkar et al., 2016) and DROP (Dua et al., 2019). For these tasks, we randomly sample 1,000 examples for training, 500 for validation, and 1,000 for testing, following the common few-shot fine-tuning protocol (Malladi et al., 2023). Performance is measured using accuracy for classification tasks and F1 scores for question answering tasks. The initialization of quantization parameters is identical to that used in quantized pre-training, and these parameters are frozen during fine-tuning. This design allows us to directly perform quantized fine-tuning without an additional quantized pre-training stage. For all fine-tuning experiments, we run our experiments three times with different seeds and report the average results.

Baselines. Besides the baseline methods used in quantized pre-training (in Section 5.1), we addi-

Table B.1: The hyperparameter for experiments. For ZeroQAT, we only show the setting of extra hyperparameters, and have the same setting in other common hyperparameters as MeZO (Malladi et al., 2023).

Experiment	Hyperparameters	Values
Quantized Pre-training	Batch size	4
	Iteration	10K
	Learning rate	{5e-7, 1e-8}
	Lr for smothing	5e-6
	Lr for clipping	1e-5
	Lr schedule	Linear Decay
Quantized Fine-tuning	ϵ in ZO	{1e-3, 5e-4 1e-4}
	Batch size	{32, 16}
	Iteration	8K
	Learning rate	{1e-6, 5e-7}
	Lr schedule	Constant
	ϵ in ZO	1e-3

tionally compare our method with several leading QAT methods, including QLoRA (Dettmers et al., 2023), QA-LoRA (Xu et al., 2023), PEQA (Kim et al., 2023), IR-QLoRA (Qin et al., 2024), EfficientQAT (Chen et al., 2025), and ZO-based method QuZO (Zhou et al., 2025).

C Ablation study on ZeroQAT

In this section, we conduct a comprehensive ablation study on ZeroQAT to illustrate the effectiveness of the components or strategies we used. Specifically, the results include:

- Effect of learnable outlier smoothing and weight clipping (Table C.1).
- Effect of using fine-tuned checkpoint by first-order as PTQ’s starting point (Table C.2).
- Effect of using lightweight ZeroQAT for quantized pre-training (Table C.3).
- Effect of the layer selection in lightweight ZeroQAT (Table C.4).
- Effect of quantize parameter initialization (Table C.5)
- Effect of the number of training samples (Table C.6).

C.1 Effect of strategies for learnable outlier mitigation

In ZeroQAT, we introduce a learnable smoothing scaling, transformation, and clipping threshold to effectively relieve the outlier issue in low-bit quantization. We conduct experiments to ablate the effectiveness of these two learnable components.

As shown in Table C.1, both components positively influence performance, but learnable smoothing proves essential for weight activation quantization. Disabling it for W4A4 results in a marked increase in perplexity, mainly due to challenges with activation quantization outliers. For weight-only quantization, smoothing only offers a slight improvement for fewer outliers occur (Shao et al., 2023), therefore, the smoothing is not used for weight-only quantization.

C.2 Effect of using first-order fine-tuned model for PTQ

When comparing our method with PTQ methods, the starting point is the full-precision fine-tuned model via ZO, therefore, we investigate if the PTQ method can perform better when using a fine-tuned model by first-order optimization. As shown in Table C.2, when using the first-order fine-tuned model as a starting point, the memory cost of fine-tuning will dramatically increase to around 100 GB, while also not enhancing the performance of PTQ, yielding much lower accuracy compared with FP ZO and ZeroQAT.

C.3 Effect of using lightweight ZeroQAT for quantized pre-training

In ZeroQAT fine-tuning, we devise a lightweight variant that keeps the query and value matrices in full precision while freezing and quantizing the remaining parameters. This design substantially reduces memory cost without sacrificing downstream task accuracy. However, when we apply the same strategy in quantized pre-training, we observe a clear performance drop, as shown in Table C.3. For example, on WikiText2, lightweight ZeroQAT

Table C.1: Effect of each component. WikiText2 perplexity is reported in this table. W/O indicates removing the corresponding learnable components. Here, smoothing indicates scaling and transformation.

PPL ↓ Learnable Components	LLama-1-7B		LLama-2-13B	
	W4A4	W2A16	W4A4	W2A16
Smoothing + Clipping	6.17	9.65	5.27	8.67
W/O Smoothing	1.4e3	29.61	5.2e3	15.97
W/O Clipping	16.64	9.4e3	18.7	2.8e3
W/O Smoothing & Clipping	2.1e3	1.2e4	1.7e4	4.6e3

Table C.2: Compare with the PTQ method with different fine-tuned models as starting points. Results of fine-tuning OPT-6.7B under W4A4 setting. ZO and FO indicate the starting fine-tuned checkpoint is from first-order and zeroth-order optimization, respectively.

Method	Fine-tuning Memory	PTQ memory	SST-2	CB	SQuAD	DROP
FP ZO	14.2 GB	-	90.2	71.4	76.0	26.4
OmniQuant (ZO)	14.2 GB	4.4 GB	61.2	48.2	24.7	11.7
OmniQuant (FO)	98.6 GB	4.4 GB	58.7	55.3	31.8	13.5
ZeroQAT	3.7 GB	-	88.6	67.4	61.3	22.4

yields perplexity of 41.05 and 21.97 for LLama2-7B and Llama2-13B under W2A16, compared to 29.61 and 15.95 without the lightweight strategy.

This degradation can be attributed to the different optimization dynamics in pre-training versus fine-tuning. Pre-training requires updating a much larger parameter space to capture broad linguistic patterns. Freezing most of the model limits the ability to adapt quantization parameters and compensate for quantization noise, leading to accumulated errors and higher perplexity. In contrast, fine-tuning operates on narrower task-specific distributions, where updating Q and V alone is sufficient to preserve performance. These results highlight that while selective fine-tuning is effective for downstream adaptation, full-parameter optimization remains crucial in the pre-training stage under quantization.

C.4 Effect of fine-tuning layer selection.

We propose a lightweight variant of ZeroQAT that fine-tunes only the query (Q) and value (V) matrices in the attention layers, while freezing and quantizing the remaining parts of the model to reduce memory overhead. To evaluate the effectiveness of this strategy, we compare it with different layer selection strategies, and the results are reported in Table C.4. The results show that this selective fine-tuning approach achieves a favorable trade-off between performance and memory efficiency: it maintains accuracy comparable to full-parameter fine-tuning, while reducing memory usage to 27%-38% of the full-parameter baseline, depending on

the model size. This demonstrates that restricting updates to Q and V matrices provides substantial efficiency gains without significant loss of performance.

C.5 Effect of initialization for smoothing parameters.

We initialize the smoothing parameters by minimizing reconstruction loss before applying ZO. To examine the impact of initialization quality, we conduct an ablation study by varying the number of initialization epochs, as reported in Table C.5. The results show that initialization has a clear effect on performance. With 0 epochs of initialization, performance drops across different models, while additional epochs (e.g., 20) can further improve accuracy. However, considering both performance gains and computational cost, we adopt two epochs as the default initialization setting. The initialization in our method is lightweight rather than computationally expensive, which takes about 5 minutes in practice, and the initialized quantization parameters can be reused across different fine-tuning tasks for the same pre-trained model. The reliance on initialization stems from the nature of ZO optimization: since ZO estimates gradients via random perturbations, starting far from a reasonable region can lead to unstable or uninformative updates, especially under low-bit quantization where the objective is highly non-smooth. Also, adapting a pre-init smoothing parameter instead of random smoothing is a common strategy in previous work (Shao et al., 2023; Sun et al., 2025).

Table C.3: Effect of using lightweight ZeroQAT in quantized pre-training. LW indicates lightweight. Perplexity on Wikitext2 is reported.

PPL ↓ Method	LLama2-7b		LLama2-13b	
	W2A16	W4A4	W2A16	W4A4
ZeroQAT (LW)	41.05	19.34	21.97	15.45
ZeroQAT	29.61	12.95	15.95	10.41

Table C.4: Ablation study for selecting which layers to maintain full-precision and update in Quantized Fine-tuning. The highlighted line with a blue rectangle is the setting used in ZeroQAT. Attn_Q: attention Query layer; Attn_V: attention Value layer; Attn_K: attention Key layer; Attn_O: attention output projection; Dense: dense fully connected layer.

Attn_Q	Attn_V	Attn_K	Attn_O	Dense	W2A16g128		W4A4	
					Acc.	Memory	Acc.	Memory
✓	✓	✓	✓	✓	60.4	100%	59.1	100%
✓	✓	✓	✓	✗	59.3	42%	58.6	50%
✓	✓	✓	✗	✗	59.4	34%	60.2	44%
✓	✓	✗	✗	✗	59.9	27%	59.4	38%
✓	✗	✗	✗	✗	48.8	20%	50.4	32%

C.6 Effect of training sample size

Conventional first-order QAT methods are generally data-inefficient, as they rely on large training datasets to provide stable and accurate gradients. To examine whether ZeroQAT exhibits similar behavior, we vary the number of training samples and report the results in Table C.6. Compared to the default setting of 128 samples, changing the sample size has only a minor effect on performance, with most perplexity variations remaining within 0.5. This indicates that, unlike conventional methods, ZeroQAT does not heavily rely on large-scale data for convergence. Instead, since its gradients are estimated through noisy zeroth-order approximations, ZeroQAT benefits more from additional optimization iterations rather than larger datasets.

D More Quantized Pre-training Results

To illustrate the generalizability of our method, we conduct quantized pre-training on OPT models, and the results are shown in Table D.2. For W6A6 quantization, similar to other baselines, ZeroQAT also achieves almost lossless results on three datasets. For a more challenging W4A4 setting, ZeroQAT consistently outperforms other baselines for better adaptation. More results on Qwen and Mistral are shown in Table D.1.

E Evaluation on MMLU and GSM8K

To demonstrate the generalizability of ZeroQAT in more realistic and challenging scenarios, we eval-

uate our method on MMLU, fine-tuning on the Alpaca dataset (Taori et al., 2023), and then evaluate. We conduct experiments based on Llama1-7B, and the results are shown in Table E.2. We also evaluate on GSM8K, and the results are shown in the Table.

F Implementation and Optimization on Mobile Device

To demonstrate the practicality of ZeroQAT in real-world deployment scenarios, we implement and evaluate our framework on a OnePlus 12 smartphone (Snapdragon 8 Gen 3) with a mobile GPU backend. Deploying LLMs on mobile devices is challenging due to strict memory and compute constraints. In particular, our device enforces a 2GB single memory pool limitation, which prevents many existing inference frameworks (e.g., llama.cpp) from directly running large-model low-bit inference such as OPT-1.3B. To address this constraint, we redesign the memory management strategy by splitting model weights into multiple smaller memory pools and carefully managing runtime buffers.

In addition, we adopt several practical engineering optimizations to improve execution efficiency on mobile hardware. These include operator fusion and execution optimizations following prior work DNNFusion (Niu et al., 2021), low-bit weight packing, and data layout optimizations inspired by SmartMem (Niu et al., 2024) to reduce memory overhead and improve memory locality. Since our

Table C.5: Effect of the number of epochs to initialize the smoothing and clipping parameter using reconstruction loss. Perplexity on WikiText2 is reported. * indicates default setting.

Epochs	LLama1-7B	LLama2-7B	OPT-6.7B
0	14.21	16.03	15.38
1	7.34	6.96	11.53
2*	6.17	5.78	11.02
10	6.06	5.71	11.17
20	5.98	5.84	10.88

Table C.6: Effect of using different numbers of training samples (token segments) for training.

Samples	W2A16	W4A4
32	19.31	6.12
64	17.88	5.96
128*	17.18	5.78
256	17.03	5.72
512	17.37	5.88

Table D.1: Results of perplexity on Wikitext2 on Qwen and Mistral.

Method	Bit	Qwen2.5-7B	Mistral-7B
-	FP16	6.67	5.54
RTN	W2A16	2.4e4	4.7e4
OmniQuant	W2A16	42.84	31.28
ZeroQAT	W2A16	18.37	14.67
OmniQuant	W4A4	15.32	13.74
FlatQuant	W4A4	7.23	5.78
ZeroQAT	W4A4	6.98	5.66

workflow performs inference-only execution without backpropagation, intermediate activations can be released after each layer finishes computation, further reducing peak memory usage. Therefore, achieving significantly low memory usage in on-device fine-tuning, as shown in Table 8.

Table D.2: Weight-activation quantization results of OPT models on three datasets: WikiText2 (WIKI), Penn Treebank (PT), and C4. RPTQ* represents a variant that quantizes all activations except the softmax output.

OPT / PPL ↓		OPT-2.7B			OPT-6.7B			OPT-13B		
Task		WIKI	PT	C4	WIKI	PT	C4	WIKI	PT	C4
FP16	-	12.47	15.13	13.16	10.86	13.09	11.74	10.13	12.34	11.20
W6A6	SmoothQuant	12.64	15.91	13.34	11.34	13.82	12.14	10.56	12.76	11.40
	RPTQ	13.19	16.37	14.04	11.19	13.98	12.08	11.19	13.98	12.08
	RPTQ*	12.71	15.53	13.33	10.96	13.24	11.86	10.96	13.24	11.86
	OmniQuant	12.62	15.32	13.29	10.96	13.20	11.81	10.21	12.47	11.17
	ZeroQAT	12.58	15.46	13.32	10.43	13.46	11.44	9.93	12.55	11.29
W4A4	SmoothQuant	131.47	107.10	120.57	1.8e4	1.4e4	1.5e4	7.4e3	6.5e3	5.6e3
	RPTQ	11.45	14.71	13.12	12.00	15.17	12.85	12.74	15.76	14.71
	RPTQ*	11.45	14.71	13.12	17.83	25.10	19.91	16.45	23.01	16.80
	OmniQuant	15.65	23.69	16.51	12.24	15.54	13.56	11.65	15.89	13.46
	ZeroQAT	13.11	18.74	14.32	11.02	13.77	12.82	10.53	13.94	12.02

Table E.1: Weight-only and weight-activation quantization results of LLama models. This table reports the accuracy of 5 zero-shot tasks.

LLama / Acc ↑	Method	PIQA	ARC-e	ARC-c	HellaSwag	Winogrande	Avg.
LLama-1-7B	FP16	77.47	72.38	41.46	73.00	67.07	65.26
	SmoothQuant	49.80	30.40	25.80	27.40	48.00	38.41
	OmniQuant	67.38	53.87	30.63	53.12	55.25	52.15
	SpainQuant	73.81	66.58	34.16	67.02	59.95	60.30
	FlatQuant	75.73	68.87	36.88	69.31	65.46	63.17
	ZeroQAT	76.98	69.32	36.19	70.86	64.87	63.64
LLama-2-7B	FP16	79.05	74.54	46.16	75.98	69.06	68.96
	SmoothQuant	54.10	31.40	24.80	29.10	50.00	37.88
	SpainQuant	75.30	65.32	39.42	71.45	63.46	62.99
	FlatQuant	77.26	72.05	43.26	73.64	69.53	67.14
	ZeroQAT	76.86	68.67	42.39	74.43	68.35	66.14
LLama-3-8B	FP16	80.74	77.57	53.50	79.12	72.93	72.77
	SmoothQuant	53.10	33.50	25.10	28.80	49.60	38.02
	SpainQuant	76.66	71.38	45.73	74.07	66.38	66.84
	FlatQuant	79.16	75.80	50.00	76.80	72.69	69.89
	ZeroQAT	79.48	77.24	51.08	77.06	71.55	71.28

Table E.2: Results of fine-tuning Llama-1-7B on MMLU benchmarks. 5-shot results are reported.

LLama-1-7B (FP: 38.41%)	GPTQ	EfficientQAT	SmoothQuant	OmniQuant	ZeroQAT
W2A16	23.71%	24.74%	-	25.65%	32.55%
W4A4	-	-	24.55%	26.93%	34.73%

Table E.3: Results of fine-tuning Llama-2-7B on GSM8K benchmarks. 5-shot results are reported.

LLama-2-7B (FP: 27.3%)	SmoothQuant	OmniQuant	FlatQuant	ZeroQAT
W2A16	-%	6.3%	18.3%	20.7%
W4A4	0.6%	7.8%	18.8%	22.1%

G Theoretical Analysis

Proposition 1 (Unbiasedness and explicit second-moment bound for the two-point ZO estimator). *Let $Q : \mathbb{R}^d \rightarrow \mathbb{Z}^d$ be the per-coordinate uniform quantizer of step size $\Delta > 0$ (rounding with optional clipping/zero-point), and let $L(\cdot; B)$ be G -Lipschitz in its argument with respect to ℓ_2 : $|L(z; B) - L(z'; B)| \leq G\|z - z'\|_2$ for all z, z' and all mini-batches B . For $\varepsilon > 0$ define the Gaussian-smoothed (forward-only) objective*

$$f_\varepsilon(W) = \mathbb{E}_{u \sim \mathcal{N}(0, I_d)} \mathbb{E}_B L(Q(W + \varepsilon u); B),$$

and the two-point ZO estimator with q i.i.d. directions $u_i \sim \mathcal{N}(0, I_d)$:

$$g_b(W; B) = \frac{1}{q} \sum_{i=1}^q \frac{L(Q(W + \varepsilon u_i); B) - L(Q(W - \varepsilon u_i); B)}{2\varepsilon} u_i.$$

Assume $\mathbb{E}_B[|L(Q(W + \varepsilon u); B)|] < \infty$ for all W and $\varepsilon > 0$. Then:

(i) **Unbiasedness.** The estimator targets the gradient of the smoothed objective:

$$\mathbb{E}_{u, B}[g_b(W; B)] = \nabla f_\varepsilon(W).$$

(ii) **Mean-squared error bound.** Writing the expectation over all randomness (u, B) ,

$$\mathbb{E} \|g_b(W; B) - \nabla f_\varepsilon(W)\|_2^2 \leq \frac{1}{q} \left[2G^2 d(d+2) + \frac{G^2 \Delta^2 d^2}{2\varepsilon^2} \right].$$

In particular, ignoring the quantizer offset term (formally $\Delta=0$), the estimator's MSE scales as $O(G^2 d^2 / q)$ under standard Gaussian directions.

Proof. (i) **Unbiasedness.** Let $U \sim \mathcal{N}(0, I_d)$ and write $Z = W + \varepsilon U$. Then $f_\varepsilon(W) = \mathbb{E}_{Z, B} L(Q(Z); B)$ with $Z \sim \mathcal{N}(W, \varepsilon^2 I_d)$. Differentiating under the integral with respect to the mean of the Gaussian and using $\nabla_W \log p_{W, \varepsilon}(Z) = (Z - W)/\varepsilon^2$,

$$\nabla f_\varepsilon(W) = \mathbb{E}_{Z, B} \left[\frac{Z - W}{\varepsilon^2} L(Q(Z); B) \right] = \frac{1}{\varepsilon} \mathbb{E}_{U, B} [U L(Q(W + \varepsilon U); B)].$$

By antithetic symmetry of U ,

$$\mathbb{E}_{U, B} \left[\frac{L(Q(W + \varepsilon U); B) - L(Q(W - \varepsilon U); B)}{2\varepsilon} U \right] = \frac{1}{\varepsilon} \mathbb{E}_{U, B} [U L(Q(W + \varepsilon U); B)],$$

hence $\mathbb{E}_{u, B}[g_b(W; B)] = \nabla f_\varepsilon(W)$.

(ii) **Second-moment/MSE bound.** Let

$$g(W; B, U) := \frac{L(Q(W + \varepsilon U); B) - L(Q(W - \varepsilon U); B)}{2\varepsilon} U.$$

Using independence of the q i.i.d. samples, $\mathbb{E} \|g_b - \nabla f_\varepsilon\|_2^2 \leq \frac{1}{q} \mathbb{E} \|g - \mathbb{E} g\|_2^2 \leq \frac{1}{q} \mathbb{E} \|g\|_2^2$. By G -Lipschitzness of $L(\cdot; B)$ and triangle inequality for Q ,

$$\|g\|_2 \leq \frac{G}{2\varepsilon} \|Q(W + \varepsilon U) - Q(W - \varepsilon U)\|_2 \|U\|_2 \leq G \|U\|_2^2 + \frac{G \Delta \sqrt{d}}{2\varepsilon} \|U\|_2,$$

where we used the standard quantization geometry $\|Q(x) - Q(y)\|_2 \leq \|x - y\|_2 + \|Q(x) - x\|_2 + \|Q(y) - y\|_2 \leq \|x - y\|_2 + \Delta \sqrt{d}$ and $\|Q(z) - z\|_2 \leq (\Delta/2) \sqrt{d}$. Applying $(a+b)^2 \leq 2a^2 + 2b^2$ and Gaussian moment identities $\mathbb{E} \|U\|_2^2 = d$, $\mathbb{E} \|U\|_2^4 = d^2 + 2d$ yields

$$\mathbb{E} \|g\|_2^2 \leq 2G^2 \mathbb{E} \|U\|_2^4 + \frac{G^2 \Delta^2 d}{2\varepsilon^2} \mathbb{E} \|U\|_2^2 = 2G^2 (d^2 + 2d) + \frac{G^2 \Delta^2 d^2}{2\varepsilon^2}.$$

Dividing by q completes the proof. $\square$

What STE assumes and why it is biased. The straight-through estimator (STE) replaces the ill-defined Jacobian $J_Q(W)$ of the piecewise-constant quantizer Q by a hand-crafted surrogate $S(W)$ (e.g., $S(W) = I$ or a clipped indicator). The chain rule then yields the surrogate update.

$$g_{\text{STE}}(W; B) = S(W)^\top \nabla_Q L(Q(W); B).$$

Because Q is flat almost everywhere, the true chain rule gives $J_Q(W) = 0$ a.e., so asserting $J_Q(W) \approx S(W)$ implicitly enforces *gradient invariance to the discrete parameterization*: $\nabla_W L(Q(W); B) \approx \nabla_Q L(Q(W); B)$ regardless of whether small perturbations of W actually change $Q(W)$. This mismatch makes g_{STE} a biased estimator of any well-defined target (e.g., $\nabla f_\varepsilon(W)$ from Gaussian smoothing, or Clarke's generalized gradient of f), and the bias can remain large away from quantization thresholds, where the true smoothed gradient vanishes in magnitude.

Proposition 2 (Worst-case STE bias in expectation, 1-D). *Assume $d = 1$ and a uniform b -bit quantizer of step $\Delta > 0$. Let $L(z; B) = Gz$ be a G -Lipschitz linear loss in its (quantized) argument. For $W \in \mathbb{R}$, let $r(W)$ be the distance to the nearest quantization threshold and set $t := r(W)/\varepsilon$. Consider the common STE choice $S(W) \equiv 1$. Then, for every W and $\varepsilon > 0$,*

$$\left\| \mathbb{E}_B[g_{\text{STE}}(W; B)] - \nabla f_\varepsilon(W) \right\| \geq G - \frac{G}{\sqrt{2\pi}} \left(\frac{\Delta}{\varepsilon} + 2t + \frac{2}{t} \right) e^{-t^2/2}.$$

In particular, for any $\delta \in (0, 1)$, if

$$t \geq \sqrt{2 \log \left(\frac{1}{\sqrt{2\pi} \delta} \left(\frac{\Delta}{\varepsilon} + 2t + \frac{2}{t} \right) \right)},$$

then $\left\| \mathbb{E}_B[g_{\text{STE}}] - \nabla f_\varepsilon(W) \right\| \geq (1 - \delta)G$; i.e., the STE exhibits an $\Omega(G)$ bias in expectation away from thresholds.

Proof via two lemmas. We first record two standard ingredients.

Lemma 1 (1-D Gaussian tail identities). *If $U \sim \mathcal{N}(0, 1)$ and $t \geq 0$, then*

$$\mathbb{E}[|U| \mathbf{1}\{|U| \geq t\}] = 2\phi(t), \mathbb{E}[U^2 \mathbf{1}\{|U| \geq t\}] = 2(t\phi(t) + 1 - \Phi(t)), \mathbb{P}(|U| \geq t) = 2(1 - \Phi(t)), \quad (5)$$

and Mills' bound $1 - \Phi(t) \leq \phi(t)/t$ holds for $t > 0$, where $\phi(t) = \frac{1}{\sqrt{2\pi}} e^{-t^2/2}$ and Φ is the standard normal cdf. $\square$

Lemma 2. *Let $d = 1$ and $t = r(W)/\varepsilon$. If $L(\cdot; B)$ is G -Lipschitz in its argument, then*

$$\left\| \nabla f_\varepsilon(W) \right\| \leq \frac{G}{\sqrt{2\pi}} \left(\frac{\Delta}{\varepsilon} + 2t + \frac{2}{t} \right) e^{-t^2/2}.$$

Proof. From the Gaussian-smoothing representation (two-point form),

$$\nabla f_\varepsilon(W) = \mathbb{E}_{u, B} \left[\frac{L(Q(W + \varepsilon u); B) - L(Q(W - \varepsilon u); B)}{2\varepsilon} u \right].$$

By G -Lipschitzness and symmetry,

$$\left\| \nabla f_\varepsilon(W) \right\| \leq \frac{G}{2\varepsilon} \mathbb{E}_u [|u| |Q(W + \varepsilon u) - Q(W - \varepsilon u)|].$$

If $|u| < t$, both perturbations stay in the same quantization cell and the difference vanishes; otherwise, the quantization geometry yields $|Q(W + \varepsilon u) - Q(W - \varepsilon u)| \leq (2\varepsilon|u| + \Delta) \mathbf{1}\{|u| \geq t\}$. Hence

$$\left\| \nabla f_\varepsilon(W) \right\| \leq \frac{G}{2\varepsilon} \mathbb{E}[(2\varepsilon|u| + \Delta)|u| \mathbf{1}\{|u| \geq t\}].$$

Expanding and applying Lemma 1 (with Mills' bound) gives the claim. $\square$

We now prove the proposition. For the stated STE with $S(W) \equiv 1$ and the linear loss $L(z; B) = Gz$, one has $\nabla_Q L(Q(W); B) \equiv G$, hence

$$\mathbb{E}_B[g_{\text{STE}}(W; B)] = G.$$

Therefore,

$$\left\| \mathbb{E}_B[g_{\text{STE}}] - \nabla f_\epsilon(W) \right\| \geq G - \|\nabla f_\epsilon(W)\| \stackrel{\text{(Lemma 2)}}{\geq} G - \frac{G}{\sqrt{2\pi}} \left(\frac{\Delta}{\epsilon} + 2t + \frac{2}{t} \right) e^{-t^2/2}.$$

Rearranging yields the thresholded $(1 - \delta)G$ lower bound.

Remark 1. *This formalizes the violation of gradient invariance and explains the $\Omega(G)$ expected bias away from thresholds. Multidimensional extensions follow by coordinate-wise threshold distances and union/tail bounds.*