First-Order Error Matters: Accurate Compensation for Quantized Large Language Models

Xingyu Zheng[1, 2*], **Haotong Qin**[3*], **Yuye Li**[4], **Haoran Chu**[2], **Jiakai Wang**[5], **Jinyang Guo**[1, 6], **Michele Magno**[3], **Xianglong Liu**[1, 2†]

[1]State Key Laboratory of Complex & Critical Software Environment, Beihang University
[2]School of Computer Science and Engineering, Beihang University
[3]ETH Zurich
[4]Xidian University
[5]Zhongguancun Laboratory
[6]School of Artificial Intelligence, Beihang University
{zhengxingyu,23371505chr,jinyangguo,xlliu}@buaa.edu.cn,
{haotong.qin,michele.magno}@pbl.ee.ethz.ch, liyueye541@gmail.com, wangjk@zgclab.edu.cn

Abstract

Post-training quantization (PTQ) offers an efficient approach to compressing large language models (LLMs), significantly reducing memory access and computational costs. Existing compensation-based weight calibration methods often rely on a second-order Taylor expansion to model quantization error, under the assumption that the first-order term is negligible in well-trained full-precision models. However, we reveal that the progressive compensation process introduces accumulated first-order deviations between latent weights and their full-precision counterparts, making this assumption fundamentally flawed. To address this, we propose **FOEM**, a novel PTQ method that explicitly incorporates first-order gradient terms to improve quantization error compensation. FOEM approximates gradients by performing a first-order Taylor expansion around the pre-quantization weights. This yields an approximation based on the difference between latent and full-precision weights as well as the Hessian matrix. When substituted into the theoretical solution, the formulation eliminates the need to explicitly compute the Hessian, thereby avoiding the high computational cost and limited generalization of backpropagation-based gradient methods. This design introduces only minimal additional computational overhead. Extensive experiments across a wide range of models and benchmarks demonstrate that FOEM consistently outperforms the classical GPTQ method. In 3-bit weight-only quantization, FOEM reduces the perplexity of Llama3-8B by 17.3% and increases the 5-shot MMLU accuracy from 53.8% achieved by GPTAQ to 56.1%. Moreover, FOEM can be seamlessly combined with advanced techniques such as SpinQuant, delivering additional gains under the challenging W4A4KV4 setting and further narrowing the performance gap with full-precision baselines, surpassing existing state-of-the-art methods.

Code — https://github.com/Xingyu-Zheng/FOEM

[*]These authors contributed equally.
[†]Corresponding author.

1 Introduction

Large language models (LLMs), such as Llama (Touvron et al. 2023), have shown remarkable performance and wide-ranging applicability in areas including language understanding (Radford et al. 2018), dialogue systems (Chen et al. 2017), code generation (Li et al. 2022), protein prediction and design (Kuhlman and Bradley 2019), and embodied intelligence (Gupta et al. 2021). As model size and the scale of pretraining data increase, their capabilities continue to improve. However, the substantial number of parameters and high computational requirements impose significant memory and processing burdens. These demands create practical limitations for real-world deployment, especially in resource-constrained environments.

Quantization is a classical model compression technique (Huang et al. 2024; Gong et al. 2024; Zheng et al. 2025; Qin et al. 2024). It reduces memory usage and speeds up computation by converting high-bit floating-point parameters and activations into low-bit fixed-point formats, without changing the model architecture. Among various quantization methods, post-training quantization (PTQ) (Lin et al. 2024; Feng et al. 2025a; Lv et al. 2024; Wang et al. 2025) is known for its efficiency. It does not require gradient-based fine-tuning and can maintain nearly lossless performance at higher bit-widths. Compared to quantization-aware training (QAT) (Gholami et al. 2022; Zheng et al. 2024a,b; Feng et al. 2025b,c,d), which involves additional training, PTQ is generally more practical for large language models. GPTQ (Frantar et al. 2022) is a representative PTQ method for weight-only quantization in large models. It estimates quantization loss using a Taylor expansion, uses second-order information from the Hessian matrix to perform column-wise quantization, and compensates for errors in later columns based on earlier quantization steps. This approach often performs better than simpler methods such as round-to-nearest (RTN).

However, we identify a potential source of error in existing LLM PTQ methods that rely on error reconstruction and compensation. These methods typically assume that the

full-precision model has already been fully optimized. Based on this assumption, they omit the first-order term in the loss modeling process. In addition, they use practical approximations to handle the second- and higher-order terms. These simplifications can lead to the accumulation of errors. As a result, the latent weights in later columns, which are calibrated after the earlier ones, may exhibit significant gradients during quantization. If these gradients are ignored, the loss modeling becomes less accurate, which can lead to suboptimal compensation and a drop in overall quantization performance.

In this work, we propose an enhanced method called **FOEM**, which compensates for output error by incorporating first-order gradient information. Instead of computing gradients through backpropagation, FOEM approximates them using the product of the Hessian matrix and the difference between the compensated latent weights and the original full-precision weights. When this approximation is substituted into the theoretical solution, the additional computation of the Hessian for the compensation term can be eliminated. This effectively avoids the high cost of real-time gradient calculation, making it feasible to integrate gradient-based correction into the PTQ process.

Extensive experiments on the Llama family demonstrate that FOEM outperforms the classic GPTQ (Frantar et al. 2022) method at any bit-width. For example, under 3-bit weight-only quantization, it reduces perplexity loss by up to 17.3%. Moreover, FOEM can be effectively combined with state-of-the-art PTQ techniques such as SpinQuant (Liu et al. 2024), further advancing the accuracy of LLM quantization. Under the challenging W4A4KV4 setting, where weights, activations, and KV cache are all quantized to 4 bits, our method further reduces the perplexity of Llama3-8B on WikiText2 (Merity et al. 2016) by 0.20. These results highlight the potential of our approach for enabling more efficient and broadly applicable deployment of large language models.

2 Related Work

Post-training Quantization. Quantization not only reduces memory consumption by mapping full-precision weights to low-bit fixed-point formats such as int8 or int4, but also enables dynamic quantization of activations into low-bit representations. This facilitates efficient operations, including multiplication of low-bit matrixes. To alleviate accuracy degradation caused by the transition from full-precision to low-bit formats, reconstruction-based methods such as AdaRound (Nagel et al. 2020), BRECQ (Li et al. 2021), and QDrop (Wei et al. 2022) have been developed. These techniques measure and minimize quantization errors within layers or blocks, demonstrating strong performance on architectures such as ResNet (He et al. 2016). However, because of the substantial computational cost incurred during calibration, these approaches are challenging to apply directly to LLMs, which typically contain billions of parameters.

PTQ Methods for LLMs. Numerous PTQ strategies have been proposed to address the outlier characteristics commonly observed in LLM. Some methods preserve outliers by maintaining them in higher-bit precision formats, such as LLM.int8() (Dettmers et al. 2022) and AWQ (Lin et al. 2023). Others employ smoothing-based scaling techniques (e.g., SmoothQuant (Xiao et al. 2023), OmniQuant (Shao et al. 2023)), rotation-based transformations (e.g., QuaRot (Ashkboos et al. 2024), SpinQuant (Liu et al. 2024)), or channel rearrangement methods (e.g., RPTQ (Yuan et al. 2023)). These approaches focus on adjusting the distributional properties of weights and activations and have demonstrated promising results in the quantization of activations. In contrast to these transformation-based methods, which typically apply scaling or clipping prior to quantization, techniques like GPTQ (Frantar et al. 2022) explicitly model the quantization loss and directly modify the fp latent weights during calibration. This loss-aware strategy can be effectively combined with other advanced quantization techniques, such as SpinQuant (Liu et al. 2024), and has recently led to significant improvements exemplified by methods like GPTAQ (Li et al. 2025b).

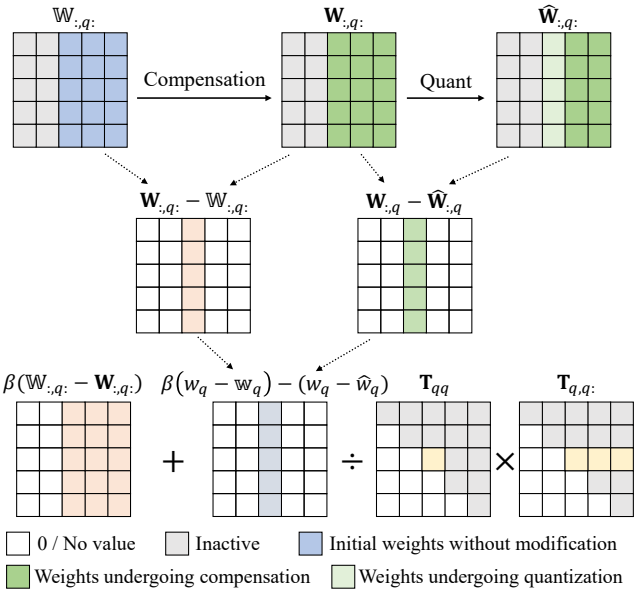

Figure 1: The computation pipeline of our proposed FOEM for the optimal compensation term with gradient consideration.

3 Method

3.1 Preliminaries

PTQ methods compensated for errors on LLM trace their origins to OBD (LeCun, Denker, and Solla 1989), a pruning technique originally developed for small-scale models. Given a layer with original weights $\mathbb{W}$ and input $\mathbf{X}$, and assuming the pruned weights are $\mathbf{W} = \mathbb{W} + \delta\boldsymbol{w}$, the resulting output error δE can be approximated using a Taylor series expansion:

$$\delta E = \left(\frac{\partial E}{\partial \boldsymbol{w}}\right)\delta\boldsymbol{w}^\top + \frac{1}{2}\delta\boldsymbol{w}\mathbf{H}\delta\boldsymbol{w}^\top + O(\|\delta\boldsymbol{w}\|^3). \quad (1)$$

OBD neglects higher-order terms and, under the assumption that the model has been well-optimized, treats the first-order term as negligible. In addition, it assumes independence

among parameters, considering only the diagonal elements of the Hessian matrix during error estimation.

OBS(Hassibi, Stork, and Wolff 1993) later challenged the independence assumption made in OBD and proposed using the full Hessian matrix to more accurately estimate the error introduced by pruning. Consider the case where the q-th parameter is pruned, and the remaining parameters are adjusted to minimize the total loss. The optimization objective can be formulated as:

$$\min_q \left\{ \min_{\delta \boldsymbol{w}} \left(\frac{1}{2} \delta \boldsymbol{w} \mathbf{H} \delta \boldsymbol{w}^\top \mid \boldsymbol{e}_q \delta \boldsymbol{w}^\top + \boldsymbol{w}_q = 0 \right) \right\}, \quad (2)$$

where q denotes the index of the weight element to be pruned, and $\boldsymbol{e}_q$ is a unit vector with a 1 at the q-th position and zeros elsewhere. Applying the method of Lagrange multipliers, the Lagrangian is defined as:

$$\mathcal{L} = \frac{1}{2} \delta \boldsymbol{w} \mathbf{H} \delta \boldsymbol{w}^\top + \lambda (\boldsymbol{e}_q \delta \boldsymbol{w}^\top + \boldsymbol{w}_q), \quad (3)$$

where λ is the Lagrange multiplier. By taking the partial derivatives with respect to both $\delta \boldsymbol{w}$ and λ, and setting them to zero, the optimal solution for $\delta \boldsymbol{w}$ can be derived as:

$$\delta \boldsymbol{w} = -\frac{\boldsymbol{w}_q}{[\mathbf{H}^{-1}]_{qq}} [\mathbf{H}^{-1}]_{q,:}. \quad (4)$$

OBC (Frantar and Alistarh 2022) observed that computing the inverse Hessian matrix after each pruning step in OBS remains prohibitively expensive for large models. To mitigate this, it restricted optimization to individual rows of the weight matrix. By considering only the second-order term in the Taylor expansion, the objective is reformulated as a sum of output losses across rows:

$$\sum_{i=1}^{d_{\text{raw}}} \left\| \mathbf{W}_{i,:} \mathbf{X} - \widehat{\mathbf{W}}_{i,:} \mathbf{X} \right\|_2^2. \quad (5)$$

It was shown that the Hessian matrix corresponding to each weight row takes the form $\mathbf{H} = 2\mathbf{X}\mathbf{X}^\top$. When a column is pruned from a given weight row, the inverse Hessian for the remaining weights can be efficiently updated through an iterative procedure:

$$\mathbf{H}_{-p}^{-1} = \left(\mathbf{H}^{-1} - \frac{1}{[\mathbf{H}^{-1}]_{qq}} \mathbf{H}_{:,p}^{-1} \mathbf{H}_{p,:}^{-1} \right)_{-p}. \quad (6)$$

This pruning-based formulation was further extended to quantization, allowing for the derivation of weight updates for the remaining columns after quantizing a specific weight element:

$$\delta \boldsymbol{w} = -\frac{\boldsymbol{w}_q - \hat{\boldsymbol{w}}_q}{[\mathbf{H}^{-1}]_{qq}} [\mathbf{H}^{-1}]_{q,:}. \quad (7)$$

GPTQ (Frantar et al. 2022) further addressed the computational inefficiencies of OBC when applied to LLMs. It was observed that the order in which the weights are quantized has a minimal effect on the final model performance, allowing it to omit both loss evaluation and sorting operations during weight selection. To improve computational efficiency, GPTQ introduced lazy updates and leveraged Cholesky decomposition. Specifically, the initial inverse Hessian $\mathbf{H}^{-1}$ is factorized as $\mathbf{H}^{-1} = \mathbf{L}\mathbf{L}^\top$, and the upper triangular matrix $\mathbf{T} = \mathbf{L}^\top$ is retained for subsequent use. This approach avoids iterative updates of $\mathbf{H}^{-1}$ during column-wise calibration and compensation, and leads to the following weight update formula during iterative quantization:

$$\delta \boldsymbol{w} = -\frac{\boldsymbol{w}_q - \hat{\boldsymbol{w}}_q}{\mathbf{T}_{qq}} \mathbf{T}_{q,q:}. \quad (8)$$

Algorithm 1: **FOEM** – First-Order Enhanced Method Based on Compensation for quantizing one layer

Input: FP weight $\mathbf{W}$, calibration input $\mathbf{X}$, and Block size B
Output: Quantized weight $\boldsymbol{Q}$
$\mathbf{H} \leftarrow \mathbf{X}\mathbf{X}^\top$
$\mathbf{L} = Inverse\_Cholesky(\mathbf{H} + \lambda_1 \boldsymbol{I})$
$\mathbb{W} \leftarrow \mathbf{W}$
$\boldsymbol{Q} \leftarrow \mathbf{0}_{m \times n}, \boldsymbol{E} \leftarrow \mathbf{0}_{m \times B}$
for $i = 0, B, 2B, \ldots$ **do**
  for $j = i, i+1, \ldots, i+B-1$ **do**
    $\boldsymbol{Q}_{:,j} \leftarrow \text{quant}(\mathbf{W}_{:,j})$
    $\boldsymbol{E}_{:,j-i} \leftarrow ((\mathbf{W}_{:,j} - \boldsymbol{Q}_{:,j}) - \beta(\mathbf{W}_{:,j} - \mathbb{W}_{:,j}))/\mathbf{L}_{jj}$
    $\mathbf{W}_{:,j:(i+B)} \leftarrow \mathbf{W}_{:,j:(i+B)} - \boldsymbol{E}_{:,j-i} \mathbf{L}_{j,j:(i+B)}^\top - \beta(\mathbf{W}_{:,j} - \mathbb{W}_{:,j})$
  end for
  $\mathbf{W}_{:,(i+B):} \leftarrow \mathbf{W}_{:,(i+B):} - \boldsymbol{E} \cdot \mathbf{L}_{i:(i+B),(i+B):}^\top$
end for

3.2 Analysis of the Neglected First-Order Term

The aforementioned methods commonly assume that well-trained models have nearly converged to local optima, thereby justifying the omission of first-order terms in the loss approximation. However, we observe that, following the quantization of preceding weight columns, the remaining unquantized weights $\mathbf{W}$ can deviate significantly from their original full-precision values $\mathbb{W}$ due to successive compensations $\delta \boldsymbol{w}$. This accumulated deviation can lead to non-negligible gradients $\boldsymbol{g} = \frac{\partial E}{\partial \boldsymbol{w}}$, making the first-order term in the Taylor expansion a potentially significant contributor to the quantized loss function.

Therefore, we begin by retaining the first-order term in the loss formulation of Eq. (1) and evaluate its impact on the final loss and compensation results. We formulate the pre- and post-quantization loss as:

$$\delta E = \boldsymbol{g} \delta \boldsymbol{w}^\top + \frac{1}{2} \delta \boldsymbol{w} \mathbf{H} \delta \boldsymbol{w}^\top. \quad (9)$$

When quantizing the q-th weight column, the optimization objective becomes minimizing the quantization-induced loss by adjusting the latent weights $\delta \boldsymbol{w}$ of the remaining unquantized columns:

$$\min_{\delta \boldsymbol{w}} \left(\boldsymbol{g} \delta \boldsymbol{w}^\top + \frac{1}{2} \delta \boldsymbol{w} \mathbf{H} \delta \boldsymbol{w}^\top \mid \boldsymbol{e}_q \delta \boldsymbol{w}^\top + \boldsymbol{w}_q - \hat{\boldsymbol{w}}_q = 0 \right), \quad (10)$$

where $\hat{\boldsymbol{w}}_q$ denotes the quantized value of the q-th weight column and $\boldsymbol{e}_q$ is a unit vector with 1 at the q-th position.

Model	Method	#W	Wiki2↓	c4↓	PiQA	Arc E	Arc C	HellaS	WinoG	BoolQ	Avg↑	MMLU↑
Llama2-7B	FP16	16	5.48	6.90	77.8	76.3	42.9	57.2	69.4	77.7	66.9	45.8
	RTN	4	5.72	7.20	77.3	75.8	42.8	56.4	68.6	77.2	66.4	44.6
	GPTQ	4	5.61	7.06	78.0	75.9	42.8	56.5	69.6	78.1	**66.8**	45.0
	GPTAQ	4	5.61	7.05	78.1	75.3	40.9	56.4	69.1	76.6	66.1	44.3
	FOEM	4	**5.61**	**7.05**	77.8	75.6	43.3	56.5	68.9	77.7	66.6	**45.5**
	RTN	3	6.92	8.66	75.5	72.5	39.2	53.8	67.5	68.0	62.7	36.1
	GPTQ	3	6.38	7.85	76.0	71.8	39.8	54.2	66.3	73.1	63.5	40.3
	GPTAQ	3	6.41	7.93	76.6	73.3	40.4	54.1	66.5	73.3	64.0	35.1
	FOEM	3	**6.27**	**7.81**	77.0	74.6	41.0	54.2	66.5	73.4	**64.5**	**42.0**
Llama2-13B	FP16	16	4.89	6.41	78.9	79.3	48.1	60.1	72.3	80.6	69.9	55.2
	RTN	4	5.03	6.58	78.8	79.7	47.5	59.8	71.0	80.3	69.5	53.6
	GPTQ	4	5.00	6.50	78.9	78.5	47.7	59.8	72.2	80.0	69.5	54.9
	GPTAQ	4	4.99	6.50	78.6	78.7	47.4	59.7	72.1	80.5	69.5	55.2
	FOEM	4	**4.99**	**6.50**	79.4	78.3	47.8	59.9	72.5	80.1	**69.7**	**55.5**
	RTN	3	5.61	7.21	77.6	77.2	43.7	56.5	69.4	76.0	66.7	49.0
	GPTQ	3	5.42	6.94	78.2	76.3	44.4	58.1	70.6	77.7	67.5	51.7
	GPTAQ	3	5.42	6.94	78.2	77.2	43.9	57.3	71.0	78.9	67.8	51.5
	FOEM	3	**5.42**	**6.93**	78.5	77.9	44.6	58.2	70.9	78.5	**68.1**	**51.8**

Table 1: Comparison of weight-only quantization for Llama-2 models.

To solve this constrained optimization problem, we apply the method of Lagrange multipliers and define the Lagrangian:

$$\mathcal{L} = \boldsymbol{g}\delta\boldsymbol{w}^\top + \frac{1}{2}\delta\boldsymbol{w}\mathbf{H}\delta\boldsymbol{w}^\top + \lambda(\boldsymbol{e}_q\delta\boldsymbol{w}^\top + \boldsymbol{w}_q - \hat{\boldsymbol{w}}_q). \quad (11)$$

Taking derivatives with respect to $\delta\boldsymbol{w}$ and λ yields:

$$\begin{cases} \dfrac{\partial\mathcal{L}}{\partial\delta\boldsymbol{w}} = \boldsymbol{g} + \delta\boldsymbol{w}\mathbf{H} + \lambda\boldsymbol{e}_q \\ \dfrac{\partial\mathcal{L}}{\partial\lambda} = \boldsymbol{e}_q\delta\boldsymbol{w}^\top + \boldsymbol{w}_q - \hat{\boldsymbol{w}}_q \end{cases}. \quad (12)$$

Setting these derivatives to zero gives the optimal $\delta\boldsymbol{w}$:

$$\delta\boldsymbol{w} = -\frac{\boldsymbol{w}_q - \hat{\boldsymbol{w}}_q - \boldsymbol{g}H^{-1}e_q^\top}{[\mathbf{H}^{-1}]_{qq}}[\mathbf{H}^{-1}]_{q,:} - \boldsymbol{g}\mathbf{H}^{-1}. \quad (13)$$

By incorporating GPTQ's Cholesky decomposition, where the inverse Hessian $\mathbf{H}^{-1} = \mathbf{L}\mathbf{L}^\top$ and $\mathbf{T} = \mathbf{L}^\top$ is retained as an upper triangular matrix, the update further simplifies to:

$$\delta\boldsymbol{w} = -\frac{\boldsymbol{w}_q - \hat{\boldsymbol{w}}_q - \boldsymbol{g}H^{-1}e_q^\top}{\mathbf{T}_{qq}}\mathbf{T}_{q,q:} - \boldsymbol{g}\mathbf{H}^{-1}. \quad (14)$$

Compared to second-order-only approaches such as Eq. (8), our formulation introduces additional terms involving g, which account for the compensation required due to previously quantized columns.

However, directly computing this term presents practical challenges: iterative updates of $\mathbf{H}^{-1}$, as described in Eq. (6), are computationally prohibitive for large language models, and obtaining the gradient $\boldsymbol{g}$ via backpropagation is similarly infeasible due to the high memory and compute cost.

3.3 Practical First-Order Error Compensation

To overcome these computational challenges, we approximate the gradient $\boldsymbol{g}$ using the difference between the full-precision weights and their compensated versions prior to quantization.

Gradient Approximation. While computing exact gradients via backpropagation after each compensation step for the remaining quantized weights $\mathbf{W}$ is computationally infeasible, we observe that the first-order gradient is closely related to the deviation between $\mathbf{W}$ and the original full-precision weights $\mathbb{W}$. By performing a Taylor expansion of the loss around $\mathbf{W}$, we can approximate the gradient on $\mathbf{W}$ as:

$$\boldsymbol{g}(\mathbf{W}) \approx \boldsymbol{g}(\mathbb{W}) + (\mathbf{W} - \mathbb{W})\mathbf{H}. \quad (15)$$

Since the initial weights $\mathbb{W}$ are assumed to be pretrained to optimality, we have:

$$\boldsymbol{g}(\mathbb{W}) \approx 0. \quad (16)$$

Therefore, we can obtain an approximate estimate of $\boldsymbol{g}(\mathbf{W})$:

$$\boldsymbol{g}(\mathbf{W}) \approx (\mathbf{W} - \mathbb{W})\mathbf{H}. \quad (17)$$

Furthermore, from a practical perspective, large gradients can significantly affect the optimization results and potentially amplify approximation errors. To address this, we introduce an empirical stabilization factor β, ensuring that the inclusion of the first-order term consistently leads to improvements. Therefore, the gradient used in practice is computed as follows:

$$g = \beta(\mathbf{W} - \mathbb{W})\mathbf{H}, \quad (18)$$

where β is typically set to 0.1, consistently leading to stable improvements.

Final Optimization Result By substituting the approximation from Eq.(18) into the theoretical expression in Eq.(14), the final compensation term, based on Eq.(8) and the Cholesky decomposition used in the original GPTQ, can be expressed as:

$$\delta\boldsymbol{w} = -\frac{(w_q - \hat{w}_q) - \beta(w_q - \mathbb{W}e_q^\top)}{\mathbf{T}_{qq}}\mathbf{T}_{q,q:} - \beta(\mathbf{W} - \mathbb{W}). \quad (19)$$

Model	Method	#W	Wiki2↓	c4↓	PiQA	Arc E	Arc C	HellaS	WinoG	BoolQ	Avg↑	MMLU↑
Llama3-8B	FP16	16	6.13	9.61	79.5	80.1	50.1	60.1	73.3	81.0	70.7	64.9
	RTN	4	6.57	10.15	79.0	79.2	47.9	59.3	73.1	80.5	69.8	63.0
	GPTQ	4	6.54	10.13	78.2	78.1	47.5	59.2	73.8	80.8	69.6	63.2
	GPTAQ	4	6.61	10.21	78.2	78.5	48.0	59.3	73.3	81.2	69.8	62.9
	FOEM	4	**6.54**	**10.13**	78.9	78.8	49.1	59.1	73.6	80.6	**70.0**	**63.2**
	RTN	3	13.10	20.50	70.3	61.2	30.9	47.4	65.1	68.0	57.2	38.9
	GPTQ	3	9.86	12.94	76.0	71.8	41.4	55.8	69.9	71.8	64.4	55.4
	GPTAQ	3	8.92	12.82	74.9	70.4	38.3	55.8	69.9	70.6	63.3	53.8
	FOEM	3	**8.32**	**12.37**	76.8	75.2	44.6	56.3	70.9	69.1	**65.5**	**56.1**
Llama3.2-1B	FP16	16	9.75	13.90	74.4	65.5	31.4	47.7	60.3	63.8	57.2	30.9
	RTN	4	11.92	17.33	72.4	60.6	30.7	45.2	57.9	57.2	54.0	28.1
	GPTQ	4	10.69	15.04	73.9	63.5	30.1	46.6	60.6	60.6	55.9	28.5
	GPTAQ	4	10.62	15.10	73.7	63.7	31.5	47.0	61.4	61.8	56.5	28.6
	FOEM	4	**10.58**	**15.02**	74.3	62.8	31.1	46.7	57.9	64.3	**56.2**	**29.0**
	RTN	3	56.41	74.73	63.4	43.6	22.5	32.6	50.0	53.5	44.3	25.5
	GPTQ	3	16.23	20.93	67.9	57.0	29.4	42.3	57.8	55.0	51.5	25.6
	GPTAQ	3	15.40	**20.46**	69.5	54.9	28.6	42.4	55.2	56.4	51.2	26.1
	FOEM	3	**15.11**	20.48	69.0	56.7	26.8	42.1	57.1	57.3	**51.5**	**26.9**
Llama3.2-3B	FP16	16	7.81	11.30	76.4	74.3	42.2	55.3	69.4	72.8	65.1	56.5
	RTN	4	8.69	12.74	76.4	71.8	42.4	53.6	68.4	72.3	64.2	51.9
	GPTQ	4	9.32	11.99	76.4	73.3	42.4	55.0	68.7	72.4	64.7	54.2
	GPTAQ	4	**8.66**	12.02	76.9	73.7	41.8	54.9	68.7	72.5	64.7	53.9
	FOEM	4	8.67	**11.98**	76.8	73.6	42.7	54.6	68.4	72.4	**64.8**	**54.4**
	RTN	3	18.43	25.38	69.1	60.2	29.9	43.3	60.9	54.7	53.0	29.8
	GPTQ	3	16.51	14.90	73.6	64.3	34.7	50.6	65.5	61.6	58.4	43.5
	GPTAQ	3	14.79	15.03	74.3	65.2	34.6	50.9	65.4	61.8	58.7	44.4
	FOEM	3	**14.16**	**14.87**	73.6	64.4	34.8	50.5	63.5	67.7	**59.1**	**46.7**

Table 2: Comparison of weight-only quantization for Llama-3 models.

Note that the computationally expensive higher-order terms in Eq. (18) and Eq. (14), including the full Hessian $\mathbf{H}$ and its inverse $\mathbf{H}^{-1}$, cancel out upon substitution. This not only removes the costly backpropagation for gradient computation but also eliminates the need to explicitly compute or invert the Hessian. Consequently, by leveraging the triangular matrices from Cholesky decomposition, only a lightweight weight difference computation is required, resulting in negligible additional overhead. Compared to the original GPTQ, the effect of our first-order correction is reflected in the additional subtracted terms in Eq. (19). Core GPTQ strategies, such as the lazy update mechanism, remain fully compatible and can be used without modification. These extensions and distinctions are summarized in Algorithm 1.

4 Experiment

We evaluated our FOEM against other advanced quantization approaches including GPTQ (Frantar et al. 2022) and GPTAQ (Li et al. 2025a) on Llama 2 (Touvron et al. 2023), Llama3 (Grattafiori et al. 2024), Qwen3 (Yang et al. 2025), Phi (Li et al. 2023) and Mistral (Jiang et al. 2023) models. In addition, we also include the baseline results of RTN(round-to-nearest) for reference. For calibration, we randomly sampled 128 data sequences from the c4 dataset with a sequence length of 2048, which is a commonly used calibration set and standard sequence length in the quantization field. The quantization configuration included weight-only quantization with group size 128 and activation quantization with KV cache quantization, where the activation quantization used the pre-learned rotation matrix published by SpinQuant by default. The evaluation tasks include perplexity (PPL) on WikiText2 (Merity et al. 2016) and C4 (Raffel et al. 2020), zero-shot accuracy on six established reasoning benchmarks (PIQA (Bisk et al. 2020), Winogrande (Sakaguchi et al. 2021), ARC-Easy (Clark et al. 2018), ARC-Challenge (Clark et al. 2018), HellaSwag (Zellers et al. 2019), and BoolQ (Clark et al. 2019)), and 5-shot MMLU (Hendrycks et al. 2020). For the single hyperparameter β in our method, we consistently set it to 0.1 as this value demonstrates robust performance across all model architectures and quantization configurations. For the coefficient α mentioned in the GPTAQ paper but not specified, we set it to 0.2 following the configuration in their official code repository. The quantization calibration process was conducted on a single NVIDIA A800-80GB GPU, while evaluating the 70B model required 2× A800 GPUs for testing. For each method and dataset, we used predefined hyperparameters and ran the experiments three times with different predetermined random seeds for data sampling, then recorded the average results.

4.1 Accuracy Results

Weight-Only Quantization. In light of GPTQ's exclusive emphasis on weight-only quantization, we first conducted a comprehensive evaluation within this paradigm. Our investi-

Model	Method	#W	Wiki2↓	c4↓	PiQA	Arc E	Arc C	HellaS	WinoG	BoolQ	Avg↑	MMLU↑
Qwen3-8B	FP16	16	7.00	10.43	79.3	81.9	52.7	58.9	72.1	83.1	71.3	76.7
	RTN	3	11.14	14.61	75.6	74.0	43.7	53.4	65.1	84.2	66.0	67.3
	GPTQ	3	8.32	11.55	77.5	78.8	50.6	55.7	68.1	76.9	68.0	67.3
	GPTAQ	3	8.20	11.54	78.5	77.0	47.3	55.5	68.1	72.5	66.5	69.6
	FOEM	3	**8.17**	**11.52**	77.3	78.1	47.1	55.6	69.2	81.7	**68.2**	**69.7**
Mistral-7B	FP16	16	5.26	7.55	80.4	80.6	50.9	61.2	74.0	83.7	71.8	62.4
	RTN	3	6.70	9.03	78.6	76.1	44.6	57.7	69.1	77.1	67.2	53.0
	GPTQ	3	6.04	8.86	79.4	78.2	47.4	58.9	72.7	80.9	69.6	55.3
	GPTAQ	3	6.05	8.85	79.8	78.2	48.3	59.1	72.5	80.9	69.8	54.5
	FOEM	3	**6.03**	**8.85**	78.9	78.3	49.5	59.4	72.4	81.5	**70.0**	**55.5**
Phi-1.5B	FP16	16	21.84	20.42	76.7	76.2	44.9	47.9	73.1	74.6	65.6	42.8
	RTN	3	27.20	23.80	74.9	72.1	42.4	45.1	69.3	71.4	62.5	36.5
	GPTQ	3	23.93	21.80	75.6	73.3	41.7	46.1	69.7	73.6	63.3	39.5
	GPTAQ	3	24.10	21.97	73.8	73.4	41.6	45.7	69.9	73.8	63.0	39.9
	FOEM	3	**23.89**	**21.72**	75.2	73.4	43.6	45.9	70.8	73.3	**63.7**	**40.1**

Table 3: Comparison of weight-only quantization for a wide variety of models.

Model	Method	Wiki2↓	c4↓	PiQA	Arc E	Arc C	HellaS	WinoG	BoolQ	Avg↑	MMLU↑
Llama2-7B	FP16	5.48	6.90	77.8	76.3	42.9	57.2	69.4	77.7	66.9	45.8
	GPTQ	6.69	8.52	74.8	71.3	39.6	51.9	64.4	72.2	62.4	35.9
	GPTAQ	6.66	8.38	74.9	73.2	38.7	52.1	63.1	71.9	62.3	36.5
	FOEM	**6.55**	**8.29**	74.6	71.5	39.5	52.7	63.8	73.2	**62.6**	**36.9**
Llama2-13B	FP16	4.89	6.41	78.9	79.3	48.1	60.1	72.3	80.6	69.9	55.2
	GPTQ	5.67	7.43	77.3	75.2	42.7	56.4	69.1	79.5	66.7	47.5
	GPTAQ	5.68	7.38	76.9	75.8	43.0	56.4	69.9	77.6	66.6	47.5
	FOEM	**5.62**	**7.36**	77.7	75.4	42.7	56.2	69.5	78.5	**66.7**	**47.5**
Llama3-8B	FP16	6.13	9.61	79.5	80.1	50.1	60.1	73.3	81.0	70.7	64.9
	GPTQ	8.55	13.24	73.6	72.3	42.0	54.1	68.0	74.6	64.1	49.6
	GPTAQ	8.50	13.13	74.5	73.2	41.6	54.5	66.5	74.2	64.1	50.4
	FOEM	**8.35**	**12.94**	74.9	72.3	41.4	54.9	67.8	74.6	**64.3**	**50.5**

Table 4: Comparison of W4A4KV4 weight–activation quantization for various Llama models, based on SpinQuant.

gation spans both 3-bit weight quantization and 4-bit quantization, enabling thorough comparisons across two widely-adopted quantization scales.

On the widely adopted Llama family of models, our approach achieves consistent improvements across all evaluation metrics, as shown in Table 1 and Table 2. For instance, in the representative case of 3-bit quantization on Llama3-8B, FOEM reduces perplexity on WikiText2 and C4 by 0.60 and 0.45 respectively, compared to the state-of-the-art compensation-based method GPTAQ. Moreover, it boosts 0-shot accuracy on commonsense datasets to 65.5 percent, narrowing the gap to the full-precision model to approximately 5 percent. In the more practically deployable 4-bit setting, FOEM consistently matches or surpasses full-precision performance. For example, on Llama2-13B, FOEM achieves parity with the fp model on the MMLU benchmark.

As shown in Table 3, FOEM also demonstrates strong performance on large language models beyond the Llama series. On Qwen3-8B (Yang et al. 2025), it reduces WikiText2 perplexity from 8.32 with GPTQ to 8.17. On Phi-1.5B (Gu and Dao 2024), FOEM is the only method capable of maintaining MMLU accuracy above 40.

Even for the large-scale Llama3-70B model, where performance remains relatively stable across different quantization methods, FOEM still achieves the best quantization results. It reduces C4 perplexity by 0.1 compared to GPTQ.

Model	Method	#W	Wiki2↓	c4↓	0-shot Avg↑
Llama3-70B	FP16	16	2.86	7.31	76.9
	GPTQ	3	5.37	9.31	73.9
	GPTAQ	3	5.41	9.26	74.0
	FOEM	3	**5.36**	**9.21**	**74.1**

Table 5: Weight-only quantization for Llama3-70B.

Weight-Activation Quantization. Our experimental analysis under W4A4KV4 configurations demonstrates the effectiveness of our method when combined with SpinQuant, the advanced rotation-based techniques for activation quantization, across multiple model scales. Notably, we directly utilize the publicly released pre-trained rotation matrices without any further tuning. For the Llama series models, our approach consistently outperforms both GPTQ and GPTAQ on the MMLU benchmark and zero-shot evaluation tasks. For example, it improves the zero-shot average accuracy of

Llama3-8B to 64.3. Although GPTQ and GPTAQ already achieve competitive performance that is close to the full-precision baseline in zero-shot settings, our method further narrows the accuracy gap and sets new state-of-the-art results. Notably, the advantage of FOEM is even more pronounced in terms of perplexity. On W3A16 quantization for Llama3-8B, it reduces ppl by approximately 0.2 compared to other advanced compensation-based methods.

Method	β	Wiki2↓	c4↓	0-shot Avg↑
GPTQ	-	9.86	12.94	64.4
FOEM	0.1	8.32	12.37	65.5
	0.2	8.87	12.90	65.7
	0.3	8.43	12.88	65.8
	0.4	8.59	12.60	64.5
	0.5	8.52	12.51	64.9
	0.6	9.67	12.72	64.6
	0.7	9.79	12.80	64.2
	0.8	10.09	13.09	61.5
	0.9	9.63	13.26	62.3
	1.0	10.37	14.41	61.5

Table 6: Sensitivity Analysis of β on Llama3-8B.

Sensitivity Analysis. We conduct a comprehensive evaluation of the impact of different settings of β on the final quantization results in Table 6. When β is less than 0.5, FOEM consistently delivers significant performance gains. Although we set β to 0.1 throughout other experiments, tuning this parameter can further enhance the effectiveness of FOEM. However, when β exceeds 0.5, the model's performance degrades substantially after quantization. This aligns with our analysis in the methodology section, where we noted that excessively large gradients can amplify approximation errors and lead to suboptimal outcomes. This finding is also consistent with observations in GPTAQ, where the use of a properly chosen stabilization coefficient is essential for ensuring consistently positive effects.

Performance on the State Space Model. Beyond standard Transformer architectures, we evaluated SSM models, exemplified by Mamba (Gu and Dao 2024). As shown in Table 7, applying W3A16 quantization to Mamba-1.4B with a default beta of 0.1, FOEM substantially outperforms GPTAQ, demonstrating strong cross-architecture generalization.

Method	β	Wikitext2 PPL↓
GPTAQ	-	14.10
FOEM	**0.1**	**13.91**
	0.2	14.06
	0.3	14.25

Table 7: Performance and sensitivity analysis of W3A16 quantization on Mamba-1.4B.

4.2 Efficiency Analysis

Compared to GPTQ, our method only introduces a simple weight difference operation that requires no matrix multiplication and adds virtually no computational overhead relative to other steps in the quantization process. As shown in Table 8, for weight-activation quantization on Llama3-8B, FOEM achieves nearly identical quantization time as GPTQ. In contrast to GPTAQ, which incurs significant additional computation and time overhead, FOEM not only offers substantially lower latency but also delivers superior accuracy. As shown in Table 9, when deploying the LLaMA3-8B model under W4A16 using vLLM (Kwon et al. 2023), we achieve over 30% speedup from weight-only quantization, in addition to the memory savings from compression. These empirical results demonstrate the effectiveness of incorporating first-order information, both in improving numerical precision and maintaining practical deployment efficiency.

Model	Method	Quant. Time (s)	Wiki2↓
Llama3-8B	GPTQ	825.50	8.55
	GPTAQ	1112.20	8.50
	FOEM	828.90	**8.35**

Table 8: Performance and Quantization Time for Llama3-8B.

Method	Input Tokens/s	Output Tokens/s
FP	184.11	470.11
FOEM	**250.26**	**616.01**

Table 9: Inference speed for Llama3-8B on W4A16.

5 Conclusion.

In this paper, we introduce FOEM, a novel PTQ method that incorporates first-order terms in the Taylor expansion of quantization loss to enable more accurate error compensation. Although full-precision models are typically assumed to be well-optimized, we observe that the application of compensation terms during quantization causes the remaining unquantized weights to deviate from their original values. As a result, the latent weights can exhibit non-negligible gradients even before quantization. To address this issue, FOEM integrates the first-order term into the Lagrangian formulation for joint optimization. Based on the derived theoretical expression, we approximate the gradient term efficiently using the difference between the original full-precision weights and the current latent weights, significantly reducing computational cost and eliminating the need for calibration data. Additionally, we utilize precomputed Cholesky factors to reconstruct the inverse Hessian submatrices on the fly, ensuring computational efficiency. FOEM consistently outperforms GPTQ across a wide range of benchmark evaluations. For example, in 3-bit weight-only quantization, FOEM reduces the perplexity of Llama3-8B by 17.3%. Furthermore, FOEM is compatible with state-of-the-art quantization strategies such as SpinQuant, offering enhanced performance while maintaining efficiency. This makes FOEM a promising solution for the practical and accurate deployment of large language models, demonstrating its broad application potential.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 62476018, 62306025, 92367204), the Fundamental Research Funds for the Central Universities, the Beijing Municipal Science and Technology Project (No. Z231100010323002), and Swiss National Science Foundation (SNSF) project 200021E_219943 Neuromorphic Attention Models for Event Data (NAMED).

References

Ashkboos, S.; Mohtashami, A.; Croci, M.; Li, B.; Cameron, P.; Jaggi, M.; Alistarh, D.; Hoefler, T.; and Hensman, J. 2024. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37: 100213–100240.

Bisk, Y.; Zellers, R.; Gao, J.; Choi, Y.; et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 7432–7439.

Chen, H.; Liu, X.; Yin, D.; and Tang, J. 2017. A survey on dialogue systems: Recent advances and new frontiers. *Acm Sigkdd Explorations Newsletter*, 19(2): 25–35.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Dettmers, T.; Lewis, M.; Belkada, Y.; and Zettlemoyer, L. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35: 30318–30332.

Feng, W.; Qin, H.; Wu, M.; Yang, C.; Li, Y.; Li, X.; An, Z.; Huang, L.; Zhang, Y.; Magno, M.; et al. 2025a. Quantized Visual Geometry Grounded Transformer. *arXiv preprint arXiv:2509.21302*.

Feng, W.; Qin, H.; Yang, C.; An, Z.; Huang, L.; Diao, B.; Wang, F.; Tao, R.; Xu, Y.; and Magno, M. 2025b. Mpq-dm: Mixed precision quantization for extremely low bit diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 16595–16603.

Feng, W.; Qin, H.; Yang, C.; Li, X.; Yang, H.; Li, Y.; An, Z.; Huang, L.; Magno, M.; and Xu, Y. 2025c. S^2Q-VDiT: Accurate Quantized Video Diffusion Transformer with Salient Data and Sparse Token Distillation. *arXiv preprint arXiv:2508.04016*.

Feng, W.; Yang, C.; Qin, H.; Li, Y.; Li, X.; An, Z.; Huang, L.; Diao, B.; Zhuang, F.; Magno, M.; et al. 2025d. Mpq-dmv2: Flexible residual mixed precision quantization for low-bit diffusion models with temporal distillation. *arXiv preprint arXiv:2507.04290*.

Frantar, E.; and Alistarh, D. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35: 4475–4488.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Gholami, A.; Kim, S.; Dong, Z.; Yao, Z.; Mahoney, M. W.; and Keutzer, K. 2022. A Survey of Quantization Methods for Efficient Neural Network Inference. In *Low-Power Computer Vision*, 291–326. Chapman and Hall/CRC.

Gong, R.; Ding, Y.; Wang, Z.; Lv, C.; Zheng, X.; Du, J.; Qin, H.; Guo, J.; Magno, M.; and Liu, X. 2024. A survey of low-bit large language models: Basics, systems, and algorithms. *arXiv preprint arXiv:2409.16694*.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gu, A.; and Dao, T. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*.

Gupta, A.; Savarese, S.; Ganguli, S.; and Fei-Fei, L. 2021. Embodied intelligence via learning and evolution. *Nature communications*, 12(1): 5721.

Hassibi, B.; Stork, D. G.; and Wolff, G. J. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, 293–299. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Huang, W.; Zheng, X.; Ma, X.; Qin, H.; Lv, C.; Chen, H.; Luo, J.; Qi, X.; Liu, X.; and Magno, M. 2024. An empirical study of llama3 quantization: From llms to mllms. *Visual Intelligence*, 2(1): 36.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; de las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; Lavaud, L. R.; Lachaux, M.-A.; Stock, P.; Scao, T. L.; Lavril, T.; Wang, T.; Lacroix, T.; and Sayed, W. E. 2023. Mistral 7B. arXiv:2310.06825.

Kuhlman, B.; and Bradley, P. 2019. Advances in protein structure prediction and design. *Nature reviews molecular cell biology*, 20(11): 681–697.

Kwon, W.; Li, Z.; Zhuang, S.; Sheng, Y.; Zheng, L.; Yu, C. H.; Gonzalez, J. E.; Zhang, H.; and Stoica, I. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

LeCun, Y.; Denker, J.; and Solla, S. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Li, Y.; Bubeck, S.; Eldan, R.; Del Giorno, A.; Gunasekar, S.; and Lee, Y. T. 2023. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Li, Y.; Choi, D.; Chung, J.; Kushman, N.; Schrittwieser, J.; Leblond, R.; Eccles, T.; Keeling, J.; Gimeno, F.; Dal Lago, A.; et al. 2022. Competition-level code generation with alphacode. *Science*, 378(6624): 1092–1097.

Li, Y.; Gong, R.; Tan, X.; Yang, Y.; Hu, P.; Zhang, Q.; Yu, F.; Wang, W.; and Gu, S. 2021. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*.

Li, Y.; Yin, R.; Lee, D.; Xiao, S.; and Panda, P. 2025a. GPTAQ: Efficient Finetuning-Free Quantization for Asymmetric Calibration. arXiv:2504.02692.

Li, Y.; Yin, R.; Lee, D.; Xiao, S.; and Panda, P. 2025b. GPTQv2: Efficient Finetuning-Free Quantization for Asymmetric Calibration. *arXiv preprint arXiv:2504.02692*.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Chen, W.-M.; Wang, W.-C.; Xiao, G.; Dang, X.; Gan, C.; and Han, S. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6: 87–100.

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

Liu, Z.; Zhao, C.; Fedorov, I.; Soran, B.; Choudhary, D.; Krishnamoorthi, R.; Chandra, V.; Tian, Y.; and Blankevoort, T. 2024. Spinquant: Llm quantization with learned rotations. *arXiv preprint arXiv:2405.16406*.

Lv, C.; Chen, H.; Guo, J.; Ding, Y.; and Liu, X. 2024. PTQ4SAM: Post-Training Quantization for Segment Anything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15941–15951.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Nagel, M.; Amjad, R. A.; Van Baalen, M.; Louizos, C.; and Blankevoort, T. 2020. Up or down? adaptive rounding for post-training quantization. In *International conference on machine learning*, 7197–7206. PMLR.

Qin, H.; Ma, X.; Zheng, X.; Li, X.; Zhang, Y.; Liu, S.; Luo, J.; Liu, X.; and Magno, M. 2024. Accurate LoRA-Finetuning Quantization of LLMs via Information Retention. *arXiv preprint arXiv:2402.05445*.

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1): 5485–5551.

Sakaguchi, K.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9): 99–106.

Shao, W.; Chen, M.; Zhang, Z.; Xu, P.; Zhao, L.; Li, Z.; Zhang, K.; Gao, P.; Qiao, Y.; and Luo, P. 2023. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Wang, J.; Zeng, Y.; Guo, J.; Ma, Y.; Liu, A.; and Liu, X. 2025. SLMQuant: Benchmarking Small Language Model Quantization for Practical Deployment. In *3rd International Workshop on Rich Media With Generative AI*.

Wei, X.; Gong, R.; Li, Y.; Liu, X.; and Yu, F. 2022. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *arXiv preprint arXiv:2203.05740*.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, 38087–38099. PMLR.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yuan, Z.; Niu, L.; Liu, J.; Liu, W.; Wang, X.; Shang, Y.; Sun, G.; Wu, Q.; Wu, J.; and Wu, B. 2023. Rptq: Reorder-based post-training quantization for large language models. *arXiv preprint arXiv:2304.01089*.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Zheng, X.; Li, Y.; Chu, H.; Feng, Y.; Ma, X.; Luo, J.; Guo, J.; Qin, H.; Magno, M.; and Liu, X. 2025. An empirical study of qwen3 quantization. *arXiv preprint arXiv:2505.02214*.

Zheng, X.; Liu, X.; Bian, Y.; Ma, X.; Zhang, Y.; Wang, J.; Guo, J.; and Qin, H. 2024a. Bidm: Pushing the limit of quantization for diffusion models. *Advances in Neural Information Processing Systems*, 37: 39009–39035.

Zheng, X.; Liu, X.; Qin, H.; Ma, X.; Zhang, M.; Hao, H.; Wang, J.; Zhao, Z.; Guo, J.; and Magno, M. 2024b. Binarydm: Accurate weight binarization for efficient diffusion models. *arXiv preprint arXiv:2404.05662*.