CAJQUANT: LARGE LANGUAGE MODEL QUANTIZATION VIA COMPENSATION-AWARE JOINT GRID SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) impose substantial memory and inference costs, motivating weight-only post-training quantization (PTQ) for efficient deployment. However, preserving low-bit accuracy while keeping quantization overhead low remains challenging. In column-wise quantization with error compensation, the candidate grid determines the quantization errors in the current column, which drive compensation updates to weights in later columns. Static group-wise grid evaluation overlooks these candidate-dependent updates by scoring grids on the weights before intra-group compensation. Moreover, clipping adjusts the spacing between grid points, but keeping the zero point fixed can overlook grids with smaller quantization errors for weights outside the clipping interval. To address these issues, we propose Compensation-Aware Joint Grid Selection for LLM Quantization (CAJQuant), which jointly selects the clipping ratio and zero point using compensation-aware grid evaluation. This evaluation measures reconstruction loss under optimal compensation while accounting for the weight updates induced by each candidate grid. Experiments on Llama models from 7B to 70B parameters demonstrate strong performance at 2, 3, and 4 bits, with substantial improvements in model quality at 2 bits. In this setting, CAJQuant outperforms all evaluated baselines in WikiText-2 perplexity and average downstream accuracy across all four models. On a single NVIDIA A100-80GB GPU, CAJQuant completes 2-bit quantization of Llama-2-70B with a group size of 128 in approximately 5 hours.

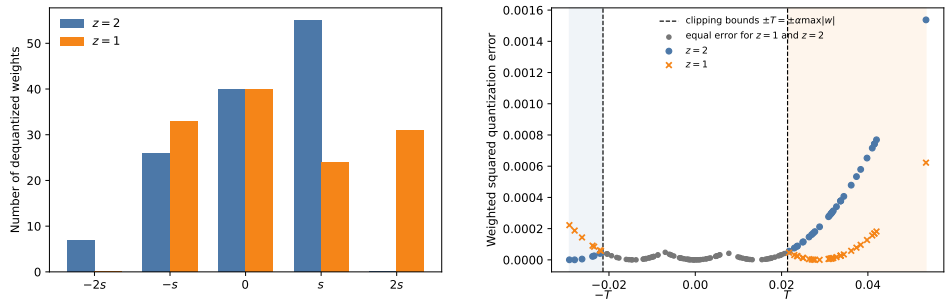
1 INTRODUCTION

Large language models (LLMs) have made substantial progress in natural language understanding, text generation, and complex reasoning (Wei et al., 2022; Kojima et al., 2022; Bubeck et al., 2023). However, the massive parameter counts lead to heavy memory usage and high computational cost, which limits deployment in resource-constrained environments (Zhang et al., 2024; Chen et al., 2023). Reducing these costs while preserving model capability is therefore essential for the widespread adoption of LLMs.

Post-training quantization (PTQ) reduces model memory requirements by converting pretrained weights to low-bit representations using a small calibration set, without large-scale retraining (Krishnamoorthi, 2018). We focus on weight-only PTQ, which compresses weights while retaining high-precision activations (Frantar et al., 2022; Lin et al., 2024; Dettmers et al., 2024). Preserving model quality in this setting requires suitable quantization grids—the sets of possible dequantized weight values—and effective compensation for quantization errors.

Existing methods address these requirements through different optimization strategies. GPTQ (Frantar et al., 2022) quantizes weights column by column and applies second-order compensation to the remaining unquantized weights. OmniQuant (Shao et al., 2024) learns weight-clipping parameters through iterative block-wise reconstruction, while LeanQuant (Zhang & Shrivastava, 2025) incorporates inverse-Hessian-derived sensitivity into grid optimization. In sequential compensation, grid selection and weight updates are tightly coupled: the candidate grid determines the quantization error in the current column, which drives compensation updates to subsequent columns and changes their

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Distribution over dequantization levels

(b) Quantization error on the same real weights

Figure 1: Effect of zero-point selection at a fixed clipping ratio. We compare 2-bit quantization of the same weight group from Llama-2-7B using a fixed scale and two zero points, $z = 1$ and $z = 2$. (a) Number of weights mapped to each dequantized value. (b) Weighted squared quantization errors for the same input weights. Dashed lines mark the clipping bounds $\pm T$. Errors are identical inside the clipping interval; outside it, $z = 2$ yields smaller errors in the negative tail, whereas $z = 1$ yields smaller errors in the positive tail.

floating-point weights before quantization. Static group-wise evaluation scores each candidate using the weights before intra-group compensation, without accounting for these candidate-dependent updates.

A grid selected using the group weights before intra-group compensation may not be well suited to the updated weights encountered during column-wise quantization. **To account for these updates, we evaluate each candidate grid using the reconstruction loss at each quantization step after optimal compensation of the remaining unquantized weights.** This loss is obtained by substituting the optimal compensation solution into the reconstruction objective. We compute it using weights updated by preceding compensation steps under the candidate grid and sum the losses over the group.

Beyond how a grid is evaluated, its treatment of weights outside the clipping interval (tail weights) also matters. For a given weight group and bit width, the clipping ratio determines the scale and thus the spacing between grid points. Once this scale is fixed, choosing between two candidate zero points determines whether the grid extends further toward the positive or negative tail. Figure 1 compares these candidates on the same weight group at a fixed clipping ratio. **In this example, the two zero-point choices yield identical dequantized values for weights inside the clipping interval, but markedly different quantization errors and corresponding reconstruction losses for weights outside it.** We therefore jointly select the clipping ratio and zero point to minimize the sum of reconstruction losses within each group.

Motivated by these considerations, we propose **CAJQuant**, a compensation-aware joint grid selection method. For each candidate clipping ratio and zero point, CAJQuant simulates column-wise quantization and error compensation within the group, starting from the same weights available at the start of the group. It selects the pair with the smallest sum of reconstruction losses. This evaluation accounts for both compensation-induced weight updates and the effect of zero-point selection on tail weights, while preserving the standard group-wise affine quantization representation.

Experiments on Llama models from 7B to 70B parameters demonstrate that CAJQuant achieves competitive perplexity and downstream accuracy at 2, 3, and 4 bits, with the largest gains at 2 bits. Ablation studies evaluate the effects of compensation-aware clipping and joint zero-point selection. On Llama-2-70B at 2 bits, CAJQuant achieves a WikiText-2 perplexity of 5.15, compared with 5.82 for LeanQuant, the strongest baseline in this setting, and completes quantization in approximately 5 hours on a single NVIDIA A100-80GB GPU.

Our main contributions are summarized as follows:

- We introduce a compensation-aware grid evaluation criterion that sums the reconstruction losses at successive steps of simulated column-wise quantization and error compensation within each group.

- We jointly select the clipping ratio and zero point under this criterion, accounting for grid spacing and the dequantized values of tail weights while preserving the standard group-wise affine representation.
- We evaluate CAJQuant across model sizes and bit widths, with ablation studies and efficiency measurements assessing its accuracy and quantization cost.

2 BACKGROUND

2.1 OBQ AND GPTQ

For a linear layer $Y = WX$, with $W \in \mathbb{R}^{m\times n}$ and calibration inputs $X \in \mathbb{R}^{n\times N}$, OBQ (Frantar & Alistarh, 2022) quantizes each output row w by minimizing its reconstruction error. For a row perturbation $\Delta w \in \mathbb{R}^{1\times n}$, the corresponding change in the layer output is ΔwX. We define the reconstruction loss as

$$L(\Delta w) = \frac{1}{2}\|\Delta wX\|_2^2 = \frac{1}{2}\Delta wH\Delta w^\top, \qquad H = XX^\top. \tag{1}$$

Following OBS (Hassibi & Stork, 1992), OBQ greedily quantizes the coordinate with the smallest optimally compensated loss and updates the remaining weights. Its row-dependent order prevents sharing inverse-Hessian updates across rows.

GPTQ (Frantar et al., 2022) instead uses a common column order and precomputes $H^{-1} = U^\top U$, where U is upper triangular. Numbering columns in this order, let $H_i = H_{i:n,i:n}$ be the Hessian on the remaining coordinates and $d_i = U_{ii}$; then $[H_i^{-1}]_{ii} = d_i^2$. We denote the current weight after preceding compensation by $\bar{w}_i$ and its dequantized value by $\tilde{w}_i$. The quantization error is the difference $\bar{w}_i - \tilde{w}_i$, which determines the compensation applied to subsequent unquantized weights. At step i, previously quantized weights remain unchanged. We fix the current weight to $\tilde{w}_i$ and optimize the adjustment to the remaining unquantized weights. The optimal update to the remaining unquantized weights is

$$\Delta w_{i:n} = \frac{\tilde{w}_i - \bar{w}_i}{[H_i^{-1}]_{ii}}\, e_i^\top H_i^{-1} = \frac{\tilde{w}_i - \bar{w}_i}{d_i^2}\,[H_i^{-1}]_{i,:}, \tag{2}$$

where e_i selects coordinate i within the remaining coordinates, and indices retain the original column numbering. This update sets the current weight to its dequantized value $\tilde{w}_i$ and adjusts the remaining unquantized weights to reduce the resulting reconstruction loss.

2.2 AFFINE QUANTIZATION AND PARAMETER SELECTION

Group-wise affine quantization (Krishnamoorthi, 2018) assigns each weight group W_g, within an output row, a shared scale and zero point. For b bits, write $K = 2^b - 1$. The quantization grid is the set of possible dequantized values, $\mathcal{G}(s, z) = \{s(k - z) : k = 0, \ldots, K\}$. Min–max quantization sets the scale to $(\max(W_g) - \min(W_g))/K$, so extreme weights can coarsen the spacing between grid points. For grid selection, let $W_g^{(0)}$ denote the weights available at the start of the group. We use a symmetric clipping interval $[-T, T]$ determined by a clipping ratio $\alpha \in (0, 1]$, with

$$T = \alpha \max_{w\in W_g^{(0)}} |w|, \qquad s_g(\alpha) = \frac{2T}{K}. \tag{3}$$

The conventional zero point is $z_g = 2^{b-1}$. The symmetric clipping interval does not assume a symmetric weight distribution. For a fixed clipping ratio, we write $s_g = s_g(\alpha)$. Using unsigned integer codes, quantization and dequantization are

$$q_i = \mathrm{clip}(\lfloor w_i/s_g \rceil + z_g,\, 0,\, K), \qquad \tilde{w}_i = Q(w_i; s_g, z_g) = s_g(q_i - z_g). \tag{4}$$

Here, q_i is the integer code, and Q denotes the quantization-and-dequantization mapping, applied elementwise to vector inputs. The conventional symmetric search fixes z_g and selects α from the group weights. Our static clipping comparator minimizes the $\ell_{2.4}$ weight error,

$$\alpha^{\ell_{2.4}} = \arg\min_{\alpha\in\mathcal{A}} \left\| Q\Big(W_g^{(0)}; s_g(\alpha), z_g\Big) - W_g^{(0)} \right\|_{2.4}^{2.4}, \qquad \mathcal{A} = \{0.21, 0.22, \ldots, 1.00\}. \tag{5}$$

This is not a squared-error objective. CAJQuant uses the same 80 clipping candidates, but evaluates them under sequential compensation and jointly searches the zero point.

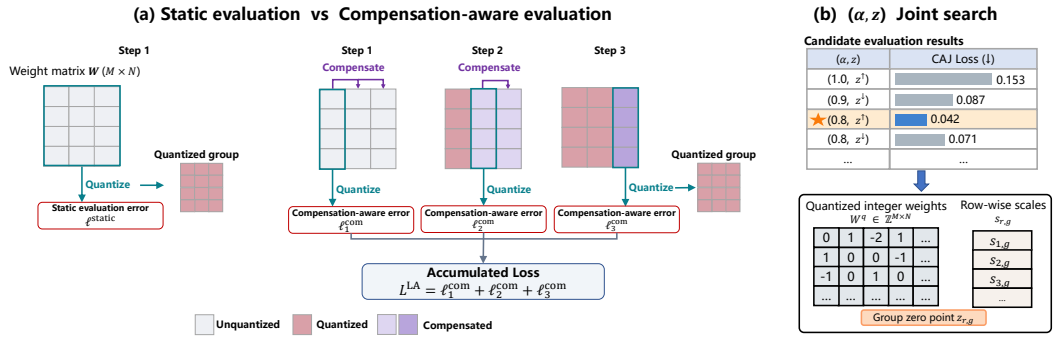


Figure 2: Overview of compensation-aware joint grid selection. (a) Static evaluation scores a candidate grid using the weights available before intra-group compensation. Compensation-aware evaluation simulates column-wise quantization and error compensation within the group, summing reconstruction loss computed from weights updated by preceding compensation steps. (b) Joint search over the clipping ratio α and zero point z selects the candidate with the smallest sum of losses. The resulting representation stores integer weight codes together with a scale $s_{r,g}$ and zero point $z_{r,g}$ for each group g in each output row r.

3 Compensation-Aware Joint Grid Selection

CAJQuant couples group-wise grid selection with GPTQ's column-wise compensation. We first formulate a grid evaluation objective that accounts for the resulting weight updates, then jointly select the clipping ratio and zero point under this objective.

3.1 Compensation-Aware Grid Evaluation

We evaluate candidate grids using the reconstruction losses computed during column-wise quantization and error compensation within each group.

At step i, let $\bar{w}_i$ denote the current weight after preceding compensation steps. For a candidate grid with scale s and zero point z, an integer code $q \in \{0, \dots, K\}$ corresponds to the dequantized value $\tilde{w}_i(q) = s(q - z)$. For each q, the optimal update $\Delta w(q)$ from Eq. (2) sets the current weight to $\tilde{w}_i(q)$ and compensates the remaining unquantized weights, while keeping previously quantized weights unchanged. Substituting this update into Eq. (1) gives the reconstruction loss $F_i(q)$ for this quantization step:

$$F_i(q) = L\big(\Delta w(q)\big) = \frac{(\tilde{w}_i(q) - \bar{w}_i)^2}{2[H_i^{-1}]_{ii}} = \frac{(\tilde{w}_i(q) - \bar{w}_i)^2}{2d_i^2}. \tag{6}$$

Let g index a group in one output row. For each candidate grid $\theta = (s, z)$, we start from the same group weights $W_g^{(0)}$ and simulate column-wise quantization and error compensation. At step i, the current weight $\bar{w}_i(\theta)$ has been updated by compensation from preceding columns under that candidate grid. We determine its integer code by minimizing $F_i(q)$ over q, which rounds and clips the updated weight onto the candidate grid:

$$q_i(\theta) = \mathrm{clip}(\lfloor \bar{w}_i(\theta)/s \rceil + z,\ 0,\ K), \qquad \tilde{w}_i(\theta) = s\big(q_i(\theta) - z\big). \tag{7}$$

Let $F_i(\theta)$ denote the reconstruction loss for step i, evaluated using the current weight $\bar{w}_i(\theta)$ and the selected integer code $q_i(\theta)$ under candidate θ. Let Θ_g denote the set of candidate grids, specified in the next subsection. We sum the reconstruction losses over the group and select the candidate with the smallest sum:

$$\begin{aligned} \mathcal{L}_g(\theta) &= \sum_{i \in g} F_i(\theta) = \sum_{i \in g} \frac{\big(\tilde{w}_i(\theta) - \bar{w}_i(\theta)\big)^2}{2d_i^2}, \\ \theta_g^* &= \arg\min_{\theta \in \Theta_g} \mathcal{L}_g(\theta). \end{aligned} \tag{8}$$

3.2 Joint Search over Clipping Ratio and Zero Point

For a given weight group and bit width, the symmetric clipping interval determines the scale through $s_g(\alpha) = 2T/K$, where the threshold T is determined by the clipping ratio α. Thus, selecting a grid $\theta = (s, z)$ reduces to selecting the clipping ratio α and zero point z.

For a given clipping ratio, the scale is determined, while the zero point controls the grid's lower and upper bounds. The conventional zero point $z^{\mathrm{sym}} = 2^{b-1}$ assigns unequal numbers of grid points to the two sides of zero. To allow the grid to extend further toward either the positive or negative tail, we consider

$$\mathcal{Z} = \{2^{b-1} - 1,\, 2^{b-1}\}. \tag{9}$$

At 2, 3, and 4 bits these candidates are $\{1, 2\}$, $\{3, 4\}$, and $\{7, 8\}$, respectively.

To isolate this boundary effect, we first compare the two zero points on the same input weight $\bar{w}_i$, holding α fixed. Using the clipping threshold T and scale $s = 2T/K$ from Section 2, write

$$n_i = \left\lfloor \frac{\bar{w}_i}{s} \right\rceil, \qquad z_1 = 2^{b-1} - 1, \qquad z_2 = 2^{b-1}. \tag{10}$$

The integer encodings and dequantized values are then

$$q_i^{(z)} = \mathrm{clip}(n_i + z,\, 0,\, K), \qquad \tilde{w}_i^{(z)} = s\,(q_i^{(z)} - z). \tag{11}$$

When $-z_1 \le n_i \le z_1$, neither encoding is clipped, so

$$\tilde{w}^{(z_1)} = \tilde{w}^{(z_2)} = s\, n_i. \tag{12}$$

When $n_i \ge z_2$ or $n_i \le -z_2$, both encodings sit on the same integer boundary, but the dequantized values differ:

$$\begin{cases} \tilde{w}_i^{(z_1)} = s\,(K - z_1), \tilde{w}_i^{(z_2)} = s\,(K - z_2) & n_i \ge z_2, \\ \tilde{w}_i^{(z_1)} = -s\, z_1, \tilde{w}_i^{(z_2)} = -s\, z_2 & n_i \le -z_2. \end{cases} \tag{13}$$

Since $z_2 = z_1 + 1$, the two reconstructions differ by one step s.

The two zero points therefore produce different dequantized values if and only if

$$\tilde{w}_i^{(z_1)} \ne \tilde{w}_i^{(z_2)} \quad \Longleftrightarrow \quad |n_i| \ge z_2. \tag{14}$$

Except at the rounding boundaries $\bar{w}_i = \pm T$, this is equivalent to

$$\frac{|\bar{w}_i|}{s} > z_2 - \frac{1}{2} = \frac{K}{2} \quad \Longleftrightarrow \quad |\bar{w}_i| > T. \tag{15}$$

Thus, for the same input weight away from the clipping boundaries, the two candidate zero points produce identical dequantized values inside the clipping interval and values that differ by one quantization step outside it (Figure 1). For a fixed clipping threshold, this step is larger at lower bit widths. The resulting quantization errors also affect subsequent weight updates, so we evaluate both zero-point candidates using the sum of reconstruction loss increments within the group.

3.3 Quantization Procedure

Algorithm 1 summarizes the procedure. At each group boundary, CAJQuant evaluates each candidate from the same weights available at the start of the group and selects (α, z) by Eq. (8). The selected grid is then used in the GPTQ pass, including the usual inter-group compensation. In Algorithm 1, $\varepsilon = (\bar{W}_{:,j} - \tilde{w})/U_{jj}$ is the quantization error normalized by U_{jj} and is used to update the remaining weights within the group.

Let G be the group size. For each candidate configuration, sequential quantization and compensation inside a group cost $\mathcal{O}(G^2)$. With $|\mathcal{A}||\mathcal{Z}|$ candidates, the search for one output row and one group therefore costs $\mathcal{O}\big(|\mathcal{A}||\mathcal{Z}|G^2\big)$. We use $|\mathcal{A}| = 80$, $|\mathcal{Z}| = 2$, and a default group size of $G = 128$.

Algorithm 1 CAJQuant for weight quantization

Require: Calibration inputs $X \in \mathbb{R}^{n \times N}$, weights $W \in \mathbb{R}^{m \times n}$, bit width b, candidates $\mathcal{A}$, block size B, group size G
Ensure: Dequantized weights $\widehat{W}$, parameters $\{s_g, z_g\}$

$H \leftarrow XX^\top$; $U \leftarrow \mathrm{CholeskyUpper}((H + \lambda I)^{-1})$
$\widehat{W} \leftarrow 0$; $\mathcal{Z} \leftarrow \{2^{b-1} - 1,\ 2^{b-1}\}$
for $i = 0, B, 2B, \ldots < n$ **do**
  $i_2 \leftarrow \min(i + B, n)$; $E \leftarrow 0_{m \times (i_2 - i)}$
  for $i_1 = i, \ldots, i_2 - 1$ **do**
    if $i_1 \bmod G = 0$ **then**
      $g \leftarrow i_1 / G$; $e \leftarrow \min(i_1 + G, n)$
      $W_g^{(0)} \leftarrow W_{:, i_1:e}$
      Compute candidate scales $s_{r,g}(\alpha)$ from $W_g^{(0)}$ for each row r
      for all $(\alpha, z) \in \mathcal{A} \times \mathcal{Z}$ **do**
        $\bar{W}_{:, i_1:e} \leftarrow W_g^{(0)}$
        $\mathcal{L}_{r,g}(\alpha, z) \leftarrow 0, \quad \forall r$
        for $j = i_1, \ldots, e - 1$ **do**
          $\tilde{w} \leftarrow Q(\bar{W}_{:,j}; s_g(\alpha), z)$
          $\varepsilon \leftarrow (\bar{W}_{:,j} - \tilde{w}) / U_{jj}$
          $\mathcal{L}_{r,g}(\alpha, z) \leftarrow \mathcal{L}_{r,g}(\alpha, z) + \frac{1}{2}\varepsilon_r^2, \quad \forall r$
          $\bar{W}_{:, j:e} \leftarrow \bar{W}_{:, j:e} - \varepsilon U_{j, j:e}$
        end for
      end for
      $(\alpha^*_{r,g}, z_{r,g}) \leftarrow \arg \min_{(\alpha,z) \in \mathcal{A} \times \mathcal{Z}} \mathcal{L}_{r,g}(\alpha, z), \quad \forall r$
      $s_{r,g} \leftarrow s_{r,g}(\alpha^*_{r,g}), \quad \forall r$
    end if
    $\widehat{W}_{:, i_1} \leftarrow Q(W_{:, i_1}; s_g, z_g)$
    $E_{:, i_1 - i} \leftarrow (W_{:, i_1} - \widehat{W}_{:, i_1}) / U_{i_1, i_1}$
    $W_{:, i_1:i_2} \leftarrow W_{:, i_1:i_2} - E_{:, i_1 - i} U_{i_1, i_1:i_2}$
  end for
  $W_{:, i_2:n} \leftarrow W_{:, i_2:n} - E U_{i:i_2, i_2:n}$
end for
return $\widehat{W}, \{s_g, z_g\}$

4 EXPERIMENTS

In this section, we conduct comprehensive experiments to assess the accuracy, quantization efficiency, and scalability of CAJQuant across multiple model families and low-bit settings. We first describe the experimental setup and then compare downstream task accuracy and language perplexity with established quantization methods. Finally, we report ablation studies and quantization efficiency results to further validate our design.

Models and quantization settings. We evaluate our method on Llama-2-7B/13B/70B (Touvron et al., 2023b) and Llama-1-30B (Touvron et al., 2023a) under W2A16, W3A16, and W4A16 weight-only quantization, with a group size of 128 for every compared method.

Baselines. We compare CAJQuant with GPTQ (Frantar et al., 2022), GPTAQ (Li et al., 2025), OmniQuant (Shao et al., 2024), and LeanQuant (Zhang & Shrivastava, 2025). For a fair comparison, we quantize the aforementioned models using their official implementations and conduct unified evaluations. LeanQuant's public code does not provide the efficient fused GPU kernel for grid learning described in its paper, so we implement that kernel following the paper. The LeanQuant times in Table 4 use this implementation.

Calibration and Evaluation. For CAJQuant, we employ a calibration dataset of 128 randomly sampled 2048-token segments from WikiText-2 (Merity et al., 2016) with a batch size of 1. The entire quantization process is performed on a single NVIDIA A100-80GB GPU. For downstream task evaluations, we report zero-shot results on PIQA (Bisk et al., 2020), ARC-Easy (Clark et al., 2018), ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), and WinoGrande (Sakaguchi et al., 2021), together with 5-shot accuracy on MMLU (Hendrycks et al., 2021). PIQA, ARC-Easy, ARC-Challenge, and HellaSwag

Table 1: Downstream task accuracy of quantized Llama models on benchmarks (higher is better).

Model	#Bits	Method	PIQA	ARC-e	ARC-c	BoolQ	HellaSwag	WinoGrande	MMLU	Avg.
2-7B	16	FP16	78.78	73.82	45.05	79.39	76.14	69.46	46.05	66.96
	4	GPTQ	77.69	70.54	43.94	77.83	75.37	70.48	44.33	65.74
	4	GPTAQ	78.24	72.60	43.69	78.78	75.22	68.27	44.45	65.89
	4	OmniQuant	78.89	72.56	44.03	78.13	75.10	68.98	45.45	66.16
	4	LeanQuant	78.35	73.27	45.22	78.04	74.71	70.01	45.29	**66.41**
	4	**CAJQuant**	78.40	73.32	45.05	77.34	75.17	69.38	45.48	66.31
	3	GPTQ	76.39	64.98	39.85	66.39	68.83	60.62	35.21	58.90
	3	GPTAQ	76.22	66.84	39.42	70.61	69.49	61.72	35.69	60.00
	3	OmniQuant	77.86	70.75	41.89	73.12	72.76	65.98	40.51	63.27
	3	LeanQuant	77.31	70.50	41.98	73.88	71.60	68.59	37.39	63.04
	3	**CAJQuant**	77.69	71.21	43.09	76.02	72.09	68.51	41.75	**64.34**
	2	GPTQ	50.16	26.52	28.67	37.71	25.80	51.38	24.16	34.91
	2	GPTAQ	49.84	26.30	26.88	46.18	25.96	49.25	24.22	35.52
	2	OmniQuant	62.08	39.44	23.98	59.66	48.10	56.35	26.65	45.18
	2	LeanQuant	67.25	48.15	27.99	64.34	49.99	61.09	24.91	49.10
	2	**CAJQuant**	66.27	49.24	29.01	65.72	49.55	59.04	26.81	**49.38**
2-13B	16	FP16	80.36	76.52	48.81	82.08	77.29	72.69	55.50	70.46
	4	GPTQ	80.03	76.05	48.81	81.56	76.85	72.14	54.70	70.02
	4	GPTAQ	80.47	75.72	48.38	81.22	76.90	71.43	54.28	69.77
	4	OmniQuant	80.47	75.46	48.29	81.74	76.39	73.01	54.76	70.02
	4	LeanQuant	79.87	76.43	48.29	81.04	76.48	72.77	54.91	69.97
	4	**CAJQuant**	79.71	76.85	49.49	82.29	76.51	73.80	54.40	**70.44**
	3	GPTQ	77.86	72.56	45.48	77.37	74.09	69.69	49.52	66.65
	3	GPTAQ	79.22	73.57	45.99	78.56	74.35	70.01	50.45	67.45
	3	OmniQuant	79.71	74.37	46.08	79.66	75.08	70.01	50.13	67.86
	3	LeanQuant	78.78	74.75	48.12	77.13	74.10	71.19	51.87	67.99
	3	**CAJQuant**	78.45	76.05	48.38	78.81	74.12	71.35	51.22	**68.34**
	2	GPTQ	47.99	27.19	29.44	38.04	25.96	49.80	26.07	34.93
	2	GPTAQ	49.84	26.09	29.44	37.83	26.14	51.38	26.38	35.30
	2	OmniQuant	63.93	45.62	29.10	63.82	52.97	54.70	27.24	48.20
	2	LeanQuant	65.02	52.90	29.35	62.39	50.00	57.30	25.54	48.93
	2	**CAJQuant**	70.73	59.43	34.81	67.25	57.57	62.04	35.16	**55.28**
1-30B	16	FP16	81.83	79.17	53.58	83.82	83.01	75.85	58.29	73.65
	4	GPTQ	81.50	78.32	52.56	83.39	82.16	76.16	57.38	73.07
	4	GPTAQ	81.99	78.32	53.75	82.66	82.14	76.95	57.71	**73.36**
	4	OmniQuant	81.23	78.49	53.33	83.30	82.34	75.30	57.89	73.13
	4	LeanQuant	81.61	78.62	52.90	82.35	82.41	75.53	57.50	72.99
	4	**CAJQuant**	81.83	78.91	53.24	82.11	82.49	76.56	57.89	73.29
	3	GPTQ	80.41	76.18	48.98	80.70	79.76	76.09	52.68	70.69
	3	GPTAQ	80.69	78.20	50.43	82.39	80.29	75.53	53.60	71.59
	3	OmniQuant	81.28	79.21	51.62	82.54	81.00	74.90	54.17	**72.10**
	3	LeanQuant	79.87	78.11	53.67	81.83	80.72	74.82	53.77	71.83
	3	**CAJQuant**	80.52	78.07	51.96	82.29	80.46	76.87	54.26	72.06
	2	GPTQ	54.73	32.70	24.91	52.23	44.81	53.20	25.34	41.13
	2	GPTAQ	58.54	35.61	24.23	52.97	45.99	52.49	24.92	42.11
	2	OmniQuant	61.59	40.87	24.83	68.41	59.80	63.85	30.21	49.94
	2	LeanQuant	74.86	67.89	38.91	70.55	67.13	71.43	32.66	60.49
	2	**CAJQuant**	75.79	67.93	40.27	67.77	69.52	70.56	36.28	**61.16**
2-70B	16	FP16	82.37	79.55	57.17	85.17	82.12	80.11	69.16	76.52
	4	GPTQ	82.21	79.38	57.00	85.14	81.58	79.01	68.18	76.07
	4	GPTAQ	81.99	79.97	57.34	84.19	81.71	79.40	68.18	76.11
	4	OmniQuant	82.21	80.35	55.97	85.20	81.86	79.32	68.35	76.18
	4	LeanQuant	82.26	80.13	57.17	84.86	81.38	79.64	68.53	**76.28**
	4	**CAJQuant**	81.83	79.55	56.74	84.43	81.60	80.03	69.13	76.19
	3	GPTQ	81.28	78.32	55.72	84.71	79.58	76.95	66.07	74.66
	3	GPTAQ	81.72	79.63	55.72	80.73	79.83	77.66	63.99	74.18
	3	OmniQuant	82.32	78.62	54.95	84.71	80.90	77.58	67.33	75.20
	3	LeanQuant	81.12	79.80	55.89	81.68	79.99	78.93	65.30	74.67
	3	**CAJQuant**	81.50	80.51	57.00	83.85	80.23	78.85	66.85	**75.54**
	2	GPTQ	65.45	46.84	32.51	64.31	49.37	60.22	26.68	49.34
	2	GPTAQ	67.63	48.86	31.14	63.49	49.07	57.14	25.87	49.03
	2	OmniQuant	75.73	67.68	40.70	70.21	68.64	69.30	44.35	62.37
	2	LeanQuant	77.37	78.07	49.83	75.26	71.36	73.64	38.01	66.22
	2	**CAJQuant**	78.02	77.86	50.77	75.44	71.75	75.93	55.68	**69.35**

use length-normalized accuracy, and BoolQ, WinoGrande, and MMLU use accuracy. In Table 1, Avg. is the unweighted mean of the accuracy scores on these seven tasks, including MMLU. All evaluations are conducted with lm-eval-harness (Gao et al., 2024). Moreover, following previous work (Frantar et al., 2022; Lin et al., 2024), we evaluate quantized LLMs using language perplexity on the WikiText-2 (Merity et al., 2016) and C4 (Raffel et al., 2020) datasets. C4 perplexity follows the OmniQuant (Shao et al., 2024) protocol of 256 random 2048-token windows from the first validation shard.

4.1 Main results

The downstream task accuracy results are presented in Table 1 and the perplexity results of quantized models are shown in Table 2. CAJQuant achieves the lowest average perplexity at all three bit widths, tying with OmniQuant at 4 bits. At 2 bits, our method reduces WikiText-2 perplexity by

Table 2: Perplexity results of Llama models under different quantization methods and bit widths (lower is better).

#Bits	Method	WikiText-2				C4				Avg.
		2-7B	2-13B	1-30B	2-70B	2-7B	2-13B	1-30B	2-70B	
FP16	-	5.473	4.885	4.103	3.319	6.973	6.470	5.980	5.524	5.341
W2	GPTQ	73.374	21.081	10.023	7.990	591.747	55.172	22.076	14.300	99.470
	GPTAQ	38.884	13.882	9.604	7.750	416.780	57.871	21.859	15.413	72.755
	OmniQuant	11.266	8.269	7.087	6.590	15.293	11.165	9.431	8.549	9.706
	LeanQuant	12.976	8.175	6.544	5.820	38.685	35.577	12.524	11.024	16.416
	CAJQuant	9.645	7.489	6.310	5.150	15.573	11.212	9.498	7.968	**9.106**
W3	GPTQ	6.412	5.400	4.748	3.860	8.554	7.311	6.654	6.015	6.119
	GPTAQ	6.160	5.328	4.673	3.856	8.380	7.239	6.603	6.038	6.035
	OmniQuant	6.095	5.321	4.592	3.796	7.811	7.023	6.393	5.873	5.863
	LeanQuant	5.937	5.184	4.456	3.667	9.321	7.036	6.367	6.092	6.007
	CAJQuant	5.929	5.200	4.462	3.640	7.781	7.011	6.381	5.824	**5.779**
W4	GPTQ	5.631	4.981	4.217	3.419	7.192	6.605	6.091	5.607	5.468
	GPTAQ	5.586	4.965	4.209	3.412	7.185	6.601	6.083	5.606	5.456
	OmniQuant	5.590	4.965	4.190	3.405	7.123	6.575	6.058	5.590	**5.437**
	LeanQuant	5.574	4.952	4.181	3.399	7.189	6.598	6.059	5.596	5.444
	CAJQuant	5.575	4.957	4.179	3.393	7.146	6.591	6.066	5.589	**5.437**

Table 3: Ablation of grid evaluation and zero-point selection on WikiText-2 (perplexity; lower is better). Underlining marks the lower perplexity between $\ell_{2.4}$ clipping and compensation-aware clipping, including ties. Bold marks the best result among all configurations for each model and bit width.

#Bits	Method	2-7B	2-13B	1-30B	2-70B
W2	No clipping search	73.374	21.081	10.023	7.990
	$\ell_{2.4}$ clipping	242.420	26.910	14.000	11.580
	Compensation-aware clipping	28.854	13.238	11.280	8.300
	CAJQuant (joint selection)	**9.645**	**7.489**	**6.310**	**5.150**
W3	No clipping search	6.412	5.400	4.748	3.860
	$\ell_{2.4}$ clipping	6.127	5.239	4.539	3.687
	Compensation-aware clipping	6.014	5.256	4.519	3.682
	CAJQuant (joint selection)	**5.929**	**5.200**	**4.462**	**3.640**
W4	No clipping search	5.631	4.981	4.217	3.419
	$\ell_{2.4}$ clipping	5.597	4.959	4.201	3.401
	Compensation-aware clipping	5.597	4.958	4.189	3.399
	CAJQuant (joint selection)	**5.575**	**4.957**	**4.179**	**3.393**

0.234 to 3.331 relative to LeanQuant across the evaluated models. At 3 and 4 bits, the perplexity remains close to the FP16 baseline of each model on each dataset, with maximum increases of 0.808 and 0.173, respectively.

In terms of downstream accuracy, CAJQuant mostly outperforms GPTQ, GPTAQ, OmniQuant, and LeanQuant. At 2 bits, the gains over OmniQuant and LeanQuant reach 11.22% on Llama-1-30B and 6.35% on Llama-2-13B, respectively. CAJQuant also achieves the highest 5-shot MMLU accuracy on all four models at 2 bits. The advantage is particularly large on Llama-2-70B, where CAJQuant reaches 55.68%, exceeding GPTAQ, OmniQuant, and LeanQuant by 29.81%, 11.33%, and 17.67%, respectively.

4.2 Ablation Study

Table 3 evaluates the effect of replacing the $\ell_{2.4}$ clipping criterion with our proposed reconstruction-loss criterion, followed by the effect of additionally searching the zero point. All clipping variants use the same 80 candidate ratios. Static $\ell_{2.4}$ clipping and compensation-aware clipping fix $z = 2^{b-1}$, whereas CAJQuant additionally searches the two zero-point candidates under the compensation-aware criterion. The configuration without clipping search (GPTQ) serves as a reference.

Q1: Does the proposed reconstruction-loss criterion improve upon $\ell_{2.4}$ clipping? We use the optional $\ell_{2.4}$ clipping criterion provided in the public GPTQ implementation (Frantar et al., 2022) as our static comparator. We replace this criterion with the sum of reconstruction losses derived from optimal compensation and evaluated at successive quantization steps using the progressively updated weights (Eq. (8)). This mostly reduces perplexity relative to static $\ell_{2.4}$ clipping, except for one tie and one small increase.

Q2: Does joint zero-point selection further decrease perplexity? Adding zero-point search to compensation-aware clipping reduces perplexity in all 12 settings. The reductions are largest at 2 bits (3.150–19.209), compared with 0.042–0.085 at 3 bits and 0.001–0.022 at 4 bits. This pattern is consistent with the analysis in Section 3: for the same input weights and a fixed clipping threshold

Table 4: Quantization time and peak memory of 2-bit weight-only PTQ. Time is measured from the start of the first layer to the end of quantization.

Method	Llama-2-7B		Llama-2-13B		Llama-1-30B		Llama-2-70B	
	Time	Mem.	Time	Mem.	Time	Mem.	Time	Mem.
GPTQ	9m	5.91 GB	18m	8.01 GB	41m	11.55 GB	1h37m	19.68 GB
LeanQuant	1h11m	5.91 GB	2h24m	8.00 GB	5h38m	11.54 GB	11h58m	19.67 GB
OmniQuant	2h31m	12.7 GB	4h31m	17.2 GB	10h35m	24.8 GB	21h29m	34.9 GB
CAJQuant	31m	7.76 GB	59m	10.09 GB	2h23m	13.82 GB	5h14m	20.32 GB

T, the two zero-point candidates produce dequantized tail values differing by one step $s = 2T/(2^b - 1)$, which is larger at lower bit widths. Joint selection evaluates these alternatives under the same reconstruction-loss criterion while accounting for the subsequent compensation updates.

4.3 Memory and time efficiency

Table 4 reports 2-bit quantization time and peak GPU memory. GPTQ is the fastest because it does not search the grid. Among methods that do, CAJQuant is faster than both LeanQuant and OmniQuant on every model. On Llama-2-70B it finishes in 5 h 14 min, compared with 11 h 58 min for LeanQuant ($2.3\times$) and 21 h 29 min for OmniQuant ($4.1\times$), while using only 0.65 GB more memory than LeanQuant (20.32 GB versus 19.67 GB). Peak memory is also well below OmniQuant (20.32 GB versus 34.9 GB).

5 Related Work

Post-training Quantization. PTQ has been widely studied in LLM compression (Frantar et al., 2022; Li et al., 2025; Kim et al., 2024; Liu et al., 2024; Sun et al., 2025). Many methods target outliers: SmoothQuant (Xiao et al., 2023) and AWQ (Lin et al., 2024) apply equivalent scaling to smooth activation outliers and protect salient weight channels, SpQR (Dettmers et al., 2024) keeps sensitive weights in higher precision, and rotation-based methods apply randomized Hadamard transforms (Tseng et al., 2024; Ashkboos et al., 2024) or learned rotations (Liu et al., 2025).

Quantization Grid Optimization. ACIQ (Banner et al., 2019) and OCTAV (Sakr et al., 2022) select clipping thresholds by minimizing MSE, and GPTQ (Frantar et al., 2022) optionally searches the clipping ratio under a weight-error norm. OmniQuant (Shao et al., 2024) learns clipping parameters by block-wise reconstruction. Among Hessian-based methods, NeUQI (Lin et al., 2025) optimizes the scale and zero point under a diagonal Hessian approximation, and LeanQuant (Zhang & Shrivastava, 2025) weights its grid objective by inverse-Hessian diagonals.

6 Conclusion

In this work, we propose CAJQuant, a fast and accurate post-training weight quantization method for LLMs. Motivated by the observation that static grid evaluation does not account for weight updates caused by sequential compensation within a group, CAJQuant evaluates each candidate grid by simulating column-wise quantization and error compensation within the group and summing the resulting reconstruction loss. Under this criterion, it jointly selects the clipping ratio and zero point, accounting for grid spacing and the effect of zero-point selection on the dequantized values of tail weights, while preserving the standard group-wise affine quantization representation. Experiments across Llama models and 2/3/4-bit settings demonstrate competitive perplexity and downstream accuracy, with substantial improvements at 2 bits. Efficiency experiments further show its ability to quantize large-scale models with low quantization overhead.

Reproducibility Statement

The compensation-aware objective and the joint search are specified in Section 3 and Algorithm 1. The clipping-ratio candidate set is defined in Section 2, and the zero-point candidates are defined in

Section 3. Models, bit widths, the calibration set, and the WikiText-2, C4, and zero-shot protocols are given in Section 4. Code and the commands used to reproduce Tables 1–4 are provided in the supplementary files.

AI USE STATEMENT

In this work, we used generative AI tools to edit it for readability, and to format it in the ICLR LaTeX template. We also used them to assist with the code that produces one figure. We have not used generative AI tools to generate data, to design the method or the experiments, to formulate the mathematical claims, or to produce or alter the reported measurements. Dataset cleaning and qualitative analysis are not applicable to this work. We have reviewed all AI-assisted text and code. The English wording was checked against the authors' technical draft, and every reported number was checked against the corresponding run log. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024.

Ron Banner, Yury Nahshan, and Daniel Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. *Advances in neural information processing systems*, 32, 2019.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. SpQR: A Sparse-Quantized Representation for Near-Lossless LLM Weight Compression. In *The Twelfth International Conference on Learning Representations*, 2024.

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

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

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, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL `https://zenodo.org/records/12608602`.

Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5, 1992.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*, 2021.

Sehoon Kim, Coleman Richard Charles Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. SqueezeLLM: Dense-and-sparse quantization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 23901–23923. PMLR, 2024.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. Gptaq: efficient finetuning-free quantization for asymmetric calibration. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25. JMLR.org, 2025.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100, 2024.

Li Lin, Xinyu Hu, and Xiaojun Wan. Neuqi: Near-optimal uniform quantization parameter initialization for low-bit llms. *arXiv preprint arXiv:2505.17595*, 2025.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. Qllm: Accurate and efficient low-bitwidth quantization for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 15417–15439, 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: Llm quantization with learned rotations. In *International Conference on Learning Representations*, volume 2025, pp. 92009–92032, 2025.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Charbel Sakr, Steve Dai, Rangha Venkatesan, Brian Zimmer, William Dally, and Brucek Khailany. Optimal clipping and magnitude-aware differentiation for improved quantization-aware training. In *International conference on machine learning*, pp. 19123–19138. PMLR, 2022.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Gao Peng, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 45472–45496, 2024.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, Wulong Liu, and Jun Yao. FlatQuant: Flatness matters for LLM quantization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 57587–57613. PMLR, 2025.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks. *Proceedings of machine learning research*, 235:48630–48656, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

Tianyi Zhang and Anshumali Shrivastava. Leanquant: Accurate and scalable large language model quantization with loss-error-aware grid. In *International Conference on Learning Representations*, volume 2025, pp. 35521–35544, 2025.

Yuxin Zhang, Lirui Zhao, Mingbao Lin, Sun Yunyun, Yiwu Yao, Xingjia Han, Jared Tanner, Shiwei Liu, and Rongrong Ji. Dynamic sparse no training: Training-free fine-tuning for sparse llms. In *International Conference on Learning Representations*, volume 2024, pp. 249–264, 2024.