REQUANT: FIXED-GRID DISCRETE REFINEMENT FOR POST-TRAINING QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training quantization (PTQ) is widely used to reduce the memory and computational cost of large language models. Existing PTQ methods typically initialize quantized models with heuristic rules or greedy optimization, and usually treat the resulting integer assignments as final. We introduce ReQuant, a backpropagation-free fixed-grid refinement stage *within* PTQ that optimizes an executable quantized model while preserving its format. Agnostic to the PTQ initializer, ReQuant starts from an existing feasible quantized model and iteratively revisits its discrete weight assignments on the fixed quantization grid. Accepted updates strictly reduce the mean squared reconstruction error and remain on the original grid. ReQuant thus integrates with existing PTQ pipelines as a plug-and-play post-processing stage. Experiments across diverse model families, bit-widths, and downstream tasks show that ReQuant consistently improves quantized models from heterogeneous PTQ initializers, with especially large gains on simple initializers and lower bit-widths. Notably, ReQuant can refine a simple round-to-nearest initialization across multiple sweeps until it approaches or surpasses GPTAQ under the same quantization format.

1 INTRODUCTION

Large language models (LLMs) based on the transformer architecture perform remarkably well across a wide range of tasks (Vaswani et al., 2017; Brown et al., 2020; Achiam et al., 2023; Liu et al., 2024a; Bai et al., 2023). However, rapid growth in LLM parameter counts substantially increases deployment costs (Touvron et al., 2023a; Dettmers et al., 2022). Modern LLMs often contain tens to hundreds of billions of parameters, with some mixture-of-experts models reaching the trillion-parameter scale, requiring substantial memory for inference (Chowdhery et al., 2023; Fedus et al., 2022; Du et al., 2022). During autoregressive decoding, model weights dominate memory consumption, making memory capacity a major bottleneck for inference.

Quantization addresses this challenge by approximating full-precision weights or activations with values from a discrete quantization grid, thereby reducing both storage and memory access costs (Jacob et al., 2018; Choi et al., 2018). As a result, quantization has become a key technique for improving deployment efficiency without modifying the model architecture (Lin et al., 2024; Xiao et al., 2023). Existing quantization methods broadly fall into two paradigms: quantization-aware training (QAT) and post-training quantization (PTQ) (Gholami et al., 2022). QAT simulates low-precision arithmetic during training with fake-quantization operations and back-propagates through the non-differentiable quantizer via the straight-through estimator (Jacob et al., 2018; Hubara et al., 2016; Bengio et al., 2013). While QAT often achieves the strongest low-bit accuracy, it requires access to training data and repeated forward–backward optimization, making it prohibitively expensive at LLM scale (Dettmers et al., 2023; Liu et al., 2024b; Chen et al., 2025; Ke et al., 2024; Malinovskii et al., 2024).

In contrast, PTQ directly quantizes pretrained full-precision models using only a small calibration set and no additional training, making it a practical alternative for large-scale deployment (Gong et al., 2025; Lin et al., 2024; Frantar et al., 2022; Li et al., 2025; Hubara et al., 2021). Existing PTQ methods mainly improve quantization by reshaping weight or activation distributions or minimizing calibration reconstruction error. Distribution-reshaping methods, such as SmoothQuant (Xiao et al., 2023), QuaRot (Ashkboos et al., 2024), and SpinQuant (Liu et al., 2024c), scale or rotate weight or activation distributions to improve their compatibility with the quantization grid. AWQ (Lin et al.,

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.

2024) instead protects salient weights from large quantization errors through activation-aware scaling. Reconstruction-based methods, such as GPTQ (Frantar et al., 2022) and GPTAQ (Li et al., 2025), build on OBQ-style second-order approximations and greedily quantize columns and compensate for quantization error.

Existing PTQ methods differ in design, yet they typically treat the grid assignments obtained after quantization as final, as shown in Figure 1. Reconstruction-based methods such as GPTQ (Frantar et al., 2022) and GPTAQ (Li et al., 2025) fix each column's discrete assignment after quantizing it and compensate for its error through the remaining columns. From a discrete optimization perspective, we can further optimize the feasible fixed-grid assignment produced by such methods.

We therefore revisit a complementary stage *within* PTQ: refining the discrete assignments of an executable quantized model without changing its deployment format. We propose ReQuant, a backpropagation-free fixed-grid refinement procedure for this stage. As a post-processing stage inside PTQ, ReQuant accepts a completed PTQ output, freezes bit-width, scales, zero-points, grid layout, and inference kernels, and directly revisits integer codes on that fixed grid. ReQuant iteratively improves quantized weights from heterogeneous PTQ initializers.

We contribute the following:

- **Method.** ReQuant formulates post-quantization discrete refinement as a composable stage inside PTQ. It treats an existing PTQ solution as a feasible initialization and optimizes its integer weight codes on the fixed quantization grid, while preserving the bit-width, scale, zero-point, grid layout, and deployment format.
- **Analysis.** We show that ReQuant reduces the overall reconstruction loss and terminates after a finite number of accepted updates, with per-sweep complexity comparable to a single GPTAQ pass.
- **Empirical results.** Across the reported model–precision pairs on Llama-3 8B/70B and Qwen3-14B, ReQuant raises mean zero-shot accuracy over four PTQ initializers, with larger gains for simple initializers and lower bit-widths. We examine refinement budgets, search radii, coordinate orders, and five QuaRot seeds to characterize cost and sensitivity. W4A16 comparisons with FlexRound (Lee et al., 2023) assess pipeline-level trade-offs, while a paired GPTQ±ReQuant control isolates refinement. A Qwen3-235B MoE experiment demonstrates large-scale feasibility with metric-dependent quality changes.

2 Related Work

Post-training quantization. Post-training quantization (PTQ) maps pre-trained weights, and optionally activations, to a discrete grid using only a small calibration set, without retraining (Banner et al., 2019; Shao et al., 2023; Xiao et al., 2023). Existing PTQ methods mainly improve quantization by either reshaping weight or activation distributions to better align with the quantization grid, or minimizing reconstruction error to preserve the behavior of the full-precision model.

Distribution-reshaping methods reshape weight or activation distributions to facilitate low-bit quantization. SmoothQuant (Xiao et al., 2023) migrates activation outliers into weights through channel-wise rescaling, thereby reducing activation quantization difficulty. OmniQuant (Shao et al., 2023) learns lightweight affine transformations to reshape weight and activation distributions during calibration. QuaRot (Ashkboos et al., 2024) and SpinQuant (Liu et al., 2024c) redistribute outliers across dimensions through orthogonal rotations, producing representations that are more robust to low-bit quantization.

Optimization-driven methods formulate PTQ as a reconstruction-error minimization problem and solve it through local or greedy optimization procedures. Building on OBQ (Frantar & Alistarh, 2022), GPTQ (Frantar et al., 2022) quantizes columns sequentially under a second-order approximation. Once a column is mapped to the quantization grid, its assignment is fixed, while the resulting quantization error is compensated by closed-form updates to the remaining unquantized columns. GPTAQ (Li et al., 2025) further accounts for the activation mismatch introduced by preceding quantized layers, using quantized-model activations to evaluate reconstruction error more accurately. In parallel, AWQ (Lin et al., 2024) takes a complementary activation-aware perspective. It identifies

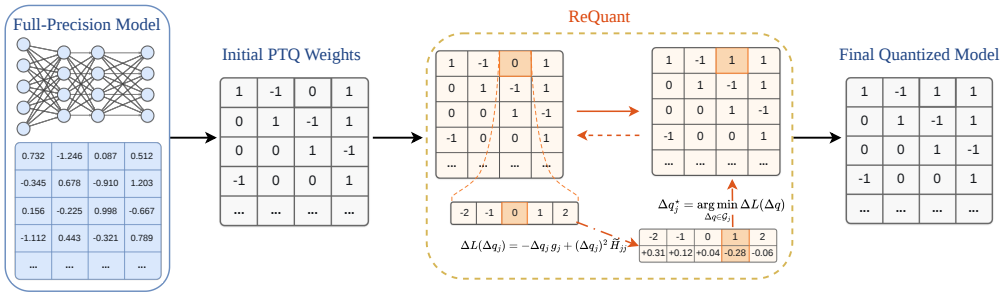


Figure 1: **Overview of ReQuant.** ReQuant refines an initial PTQ model by iteratively updating discrete weight assignments on the fixed quantization grid, improving reconstruction error while preserving the quantized format.

salient weight channels using activation statistics and rescales channels to better preserve their contribution after quantization.

Construction-time continuous relaxation vs. post-quantization fixed-grid refinement. A related but distinct line of work optimizes quantization decisions while *constructing* the quantized model from full-precision weights. AdaRound (Nagel et al., 2020) learns continuous rounding variables through gradient-based optimization; BRECQ (Li et al., 2021) extends reconstruction-based PTQ to block-level settings; and FlexRound (Lee et al., 2023) jointly learns element-wise division factors and a quantization-grid scale via STE-based backpropagation for Transformers. AdaQuant (Hubara et al., 2021) similarly optimizes layer-level reconstruction with continuous parameters. These methods introduce continuous surrogate variables, optimize a relaxed objective, and then discretize to obtain the final codes. They share a reconstruction objective with ReQuant, but operate at a different pipeline stage and optimize different variables.

ReQuant instead starts *after* an upstream PTQ method produces an executable quantized model. It freezes the inherited bit-width, scales, zero-points, quantization grid, storage format, and inference kernels, and directly updates integer codes on that fixed grid with exact discrete loss-change evaluations, remaining free of model backpropagation, STE, and optimizer states for learnable quantizer parameters. Consequently, every intermediate solution remains deployable under the original format. We therefore position ReQuant as a complementary, initializer-agnostic refinement stage inside PTQ that can be composed with AdaRound/BRECQ/FlexRound-style construction-time optimization as well as with standard heuristic or greedy PTQ pipelines.

3 Method

3.1 Preliminaries

For a linear layer with full-precision weights $\mathbf{W} \in \mathbb{R}^{d_{\text{row}} \times d_{\text{col}}}$ and calibration activations $\mathbf{X} \in \mathbb{R}^{d_{\text{col}} \times m}$, classical layer-wise PTQ is commonly formulated as minimizing the reconstruction error

$$\min \ \|\mathbf{W}\mathbf{X} - \mathbf{W}_q\mathbf{X}\|_F^2. \tag{1}$$

However, at inference time, the input to a given layer is produced by preceding quantized layers rather than by the full-precision model. To model this activation mismatch, GPTAQ (Li et al., 2025) replaces the full-precision activations $\mathbf{X}$ in the quantized branch with the activations $\widetilde{\mathbf{X}}$ observed under the quantized prefix, yielding the activation-aware objective

$$\min \ \mathcal{L}(\mathbf{W}_q) \ = \ \|\mathbf{W}\mathbf{X} - \mathbf{W}_q\widetilde{\mathbf{X}}\|_F^2. \tag{2}$$

Both $\mathbf{X}$ and $\widetilde{\mathbf{X}}$ can be collected by forward passes on the calibration set, using the full-precision and partially quantized models, respectively.

Notation. For a matrix $\mathbf{W} \in \mathbb{R}^{d_{\text{row}} \times d_{\text{col}}}$, the first subscript indexes rows and the second indexes columns: $\mathbf{W}_{i,:}$ denotes its i-th row and $\mathbf{W}_{:,j}$ denotes its j-th column. For an ordered index set R, $\mathbf{W}_{i,R}$ denotes the entries of row i restricted to the columns in R, and $\mathbf{W}_{R,R}$ denotes the corresponding principal submatrix when the matrix is square. The same convention applies to $\widetilde{\mathbf{H}} := \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^\top$ and other matrices appearing below.

3.2 ReQuant Method

Figure 1 illustrates ReQuant. Existing PTQ methods typically stop after a single quantization procedure and treat the resulting on-grid assignment as final, whereas ReQuant starts from this assignment and further optimizes discrete weights on the same grid.

Optimization objective. With the quantization parameters (e.g., scale and zero-point) fixed, we optimize the quantized weights over the induced discrete grid:

$$\min_{\mathbf{W}_q \in \mathcal{G}} \mathcal{L}(\mathbf{W}_q) \;=\; \|\mathbf{W}\mathbf{X} - \mathbf{W}_q\widetilde{\mathbf{X}}\|_F^2. \tag{3}$$

where $\mathcal{G}$ is the fixed quantization grid. We refer to $\mathcal{L}(\mathbf{W}_q)$ as the layer-wise reconstruction objective.

Row-wise decomposition. Each weight row independently produces the corresponding output row of a linear layer. In particular, the i-th row of $\mathbf{W}_q$ only affects the i-th row of the quantized output $\mathbf{W}_q\widetilde{\mathbf{X}}$. Therefore, the reconstruction loss decomposes as

$$\|\mathbf{W}\mathbf{X} - \mathbf{W}_q\widetilde{\mathbf{X}}\|_F^2 = \sum_{i=1}^{d_{\text{row}}} \|\mathbf{W}_{i,:}\mathbf{X} - \mathbf{W}_{q,i,:}\widetilde{\mathbf{X}}\|_2^2. \tag{4}$$

Updating a single coordinate $W_{q,i,j}$ affects only the i-th term in Eq. 4; the losses of all other rows remain unchanged. Thus, reducing L for any single row directly decreases the layer-level loss $\mathcal{L}(\mathbf{W}_q)$, and ReQuant can refine each row independently and in parallel. In the rest of this section we derive the update rule for a single row and omit the row index for clarity.

Row-level reconstruction loss. Following the row-wise decomposition above, we focus on a single row and omit the row index. Let $\mathbf{w} = \mathbf{W}_{i,:}$ denote the full-precision weight row, $\mathbf{q} = \mathbf{W}_{q,i,:}$ denote its current quantized counterpart, and $\mathbf{e} = \mathbf{w} - \mathbf{q}$ denote the corresponding quantization error. By Eq. 4, the contribution of this row to $\mathcal{L}(\mathbf{W}_q)$ is

$$L(\mathbf{e}) = \|\mathbf{w}\mathbf{X} - \mathbf{q}\widetilde{\mathbf{X}}\|_2^2 = \|\mathbf{e}\widetilde{\mathbf{X}} - \mathbf{w}\Delta\mathbf{X}\|_2^2, \tag{5}$$

where $\Delta\mathbf{X} = \widetilde{\mathbf{X}} - \mathbf{X}$ denotes the activation mismatch induced by earlier quantized layers. Eq. 5 rewrites the row-level loss as a function of the quantization error $\mathbf{e}$, which is the form on which ReQuant operates.

We define $\mathbf{g}$ as the gradient of L with respect to $\mathbf{e}$:

$$\mathbf{g} = \nabla_{\mathbf{e}} L(\mathbf{e}) = 2(\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B}), \tag{6}$$

where $\mathbf{B} = \Delta\mathbf{X}\widetilde{\mathbf{X}}^\top$ and $\widetilde{\mathbf{H}} = \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^\top \succeq 0$ is the activation Gram matrix. The Hessian of L with respect to $\mathbf{e}$ is $2\widetilde{\mathbf{H}}$, so L is convex and its curvature along coordinate j is $2\widetilde{H}_{jj}$. If the quantized value at coordinate j is moved by an on-grid step Δq_j, then the quantization error becomes $\mathbf{e}' = \mathbf{e} - \Delta q_j \mathbf{u}_j$, and the loss change is

$$\Delta L(\Delta q_j) = L(\mathbf{e}') - L(\mathbf{e}) = -\Delta q_j \, g_j + (\Delta q_j)^2 \, \widetilde{H}_{jj}. \tag{7}$$

We derive this result in Appendix A.1. Once the current row loss L and vector $\mathbf{g}$ are computed, Eq. 6 and Eq. 7 directly evaluate subsequent coordinate updates. ReQuant therefore maintains only L and $\mathbf{g}$ during refinement to reduce candidate-evaluation cost.

Algorithm 1 Post-Quantization Discrete Coordinate Refinement

Input: full-precision row $\mathbf{w}$, initial quantized row $\mathbf{q}$, grid information, $\widetilde{\mathbf{H}}$, $\mathbf{B}$, search radius K, sweeps T
$\mathbf{e} \leftarrow \mathbf{w} - \mathbf{q}, \quad \mathbf{g} \leftarrow 2(\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B})$
$L \leftarrow \|\mathbf{e}\widetilde{\mathbf{X}} - \mathbf{w}\Delta\mathbf{X}\|_2^2$
for $t = 1, \ldots, T$ **do**
  for $j = 1, \ldots, d_{\text{col}}$ **do**
    $\Delta q_j^\star \leftarrow \arg\min_{\Delta q_j \in \mathcal{G}_j^{(K)}} \Delta L(\Delta q_j)$
    if $\Delta L(\Delta q_j^\star) < 0$ **then**
      $q_j \leftarrow q_j + \Delta q_j^\star$
      $\mathbf{g} \leftarrow \mathbf{g} - 2\Delta q_j^\star \widetilde{\mathbf{H}}_{j,:}$
      $L \leftarrow L + \Delta L(\Delta q_j^\star)$
    end if
  end for
end for
Return: refined quantized row $\mathbf{q}$

Coordinate update. Let s_j, o_j, and z_j denote the scale, zero-point, and quantized integer code of coordinate j, so that $q_j = (z_j - o_j)s_j$. Starting from the current integer code z_j, ReQuant considers integer offsets k such that the updated code $z_j + k$ remains within the representable range. Therefore, any candidate update has the form $\Delta q_j = ks_j$, and the feasible set at coordinate j is

$$\mathcal{G}_j = \{\, ks_j : k \in \mathbb{Z},\ k \neq 0,\ z_{\min} \leq z_j + k \leq z_{\max} \,\}. \tag{8}$$

We restrict integer offsets to a search radius K,

$$\mathcal{G}_j^{(K)} = \{\, ks_j \in \mathcal{G}_j : 1 \leq |k| \leq K \,\},$$

giving at most $2K$ feasible nonzero moves, with fewer near code boundaries. We score candidates with Eq. 7, selecting

$$\Delta q_j^\star = \arg \min_{\Delta q \in \mathcal{G}_j^{(K)}} \Delta L(\Delta q). \tag{9}$$

This step applies discrete coordinate descent. With all other coordinates fixed, ReQuant moves coordinate j to the best feasible grid point in the K-neighborhood.

ReQuant accepts a move whenever it strictly decreases the row loss, i.e., $\Delta L(\Delta q_j^\star) < 0$. Upon acceptance, ReQuant updates coordinate j and increments the row loss by $L \leftarrow L + \Delta L(\Delta q_j^\star)$. ReQuant then updates the gradient vector incrementally:

$$\mathbf{g} \leftarrow \mathbf{g} - 2\Delta q_j^\star \widetilde{\mathbf{H}}_{j,:}. \tag{10}$$

This follows from $\mathbf{g} = 2(\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B})$, since an accepted move changes $\mathbf{e}$ only at coordinate j. Therefore, $\mathbf{g}$ can be updated in $O(d_{\text{col}})$ time using the j-th row of $\widetilde{\mathbf{H}}$.

Because each accepted update changes $\mathbf{e}$ and hence the score vector $\mathbf{g}$, a coordinate that appears locally fixed in one pass can become improvable after other coordinates have changed. Thus, a single greedy traversal over $j = 1, \ldots, d_{\text{col}}$ leaves additional discrete improvements available for the coupled quadratic objective $L(\mathbf{e})$. ReQuant therefore repeats the coordinate cycle for a fixed number of sweeps T, as summarized in Algorithm 1, providing a controllable compute–accuracy trade-off.

3.3 Analysis

Convergence. ReQuant performs discrete coordinate descent over a finite quantization grid, searching the K-neighborhood $\mathcal{G}_j^{(K)}$ at each coordinate. At each step, a candidate update is accepted whenever $\Delta L(\Delta q_j^\star) < 0$ (Algorithm 1). Hence, every accepted update strictly decreases the corresponding row-level loss L. Since the layer objective decomposes as $\mathcal{L}(\mathbf{W}_q) = \sum_i L(\mathbf{e}_i)$ and each update affects a single row, it also strictly decreases $\mathcal{L}(\mathbf{W}_q)$. Strict descent ensures that each accepted move reaches a distinct quantized state in the finite feasible set. If refinement continues until every K-neighborhood move has been exhausted, the returned solution is a coordinate-wise local optimum within that neighborhood; in practice a fixed budget of T sweeps already yields strong empirical

improvements while keeping offline cost controllable. Appendix A.2 proves the infinite-sweep result in full.

A larger, representative calibration set may yield more reliable estimates of the activation statistics and reconstruction objective. This does not guarantee that coordinate-local refinement approaches a population-level optimum or that downstream metrics improve monotonically; we assess generalization empirically on held-out data.

Reconstruction loss versus downstream metrics. The analysis guarantees monotone decrease of the empirical layer-wise objective along accepted updates, and neighborhood-local optimality when the K-search is run to exhaustion. Reducing layer reconstruction error is a standard proxy in PTQ (Nagel et al., 2020; Li et al., 2021; Frantar et al., 2022): if the remaining network is locally Lipschitz, smaller layer residuals tighten a bound on later hidden-state perturbations. Consistent with this proxy, our experiments show that reconstruction-driven refinement typically improves held-out PPL, KL, and zero-shot accuracy, which we report as empirical outcomes of Algorithm 1.

Efficiency. ReQuant evaluates candidate updates from cached statistics rather than recomputing the full reconstruction loss each time. Using the row-wise decomposition and the closed-form coordinate score in Eq. 7, it scores on-grid moves efficiently and refreshes the score vector after each accepted move with a rank-one update. ReQuant can refine independent output rows in parallel using shared activation statistics. For a layer with T refinement sweeps, the dominant cost is $O(md_{\text{col}}^2 + (T+1)d_{\text{row}}d_{\text{col}}^2)$ under low-bit quantization, where each coordinate has a small number of feasible moves. Appendix A.3 analyzes the computational complexity in full.

4 Experiments

4.1 Experimental Setup

We evaluate Llama-3 (Grattafiori et al., 2024) and Qwen3 (Yang et al., 2025) across PTQ initializers, precisions, refinement budgets, and model scales up to 235B parameters.

Implementation details. We implement ReQuant with Hugging Face Transformers (Wolf et al., 2020) and PyTorch (Paszke et al., 2019). We collect activation statistics from 512 WikiText-2 (Merity et al., 2016) calibration sequences of length 2048. Unless otherwise specified, we asymmetrically quantize weights per channel and activations per tensor. Following GPTAQ (Li et al., 2025), activation quantizers are calibrated and applied before weight quantization, so ReQuant optimizes weights under the activations produced by the quantized model. By default, we set the number of ReQuant refinement sweeps to $T{=}4$ and evaluate $K{=}2$ neighboring grid moves in each direction for every coordinate. Unless noted otherwise, main tables use data-parallel calibration on eight NVIDIA GeForce RTX 4090 GPUs; Llama-3 70B uses four NVIDIA B200 GPUs. FlexRound comparisons (Section 4.5) are timed on an NVIDIA RTX PRO 6000 for memory-matched fairness, and the Qwen3-235B experiment (Section 4.6) uses eight NVIDIA H200 GPUs.

Evaluation. We evaluate language modeling performance by reporting perplexity on WikiText-2 (Merity et al., 2016), UltraChat-2k (Ding et al., 2023), and NuminaMath (Li et al., 2024). For downstream evaluation, we report accuracy on ten standard commonsense reasoning and language understanding benchmarks, including ARC-Challenge (ARC-C), ARC-Easy (ARC-E), BoolQ, CEval, HellaSwag, LAMBADA, OpenBookQA (OBQA), PIQA, SocialIQA (SIQA), and Winogrande (Clark et al., 2018; 2019; Huang et al., 2023; Zellers et al., 2019; Paperno et al., 2016; Mihaylov et al., 2018; Bisk et al., 2020; Sap et al., 2019; Sakaguchi et al., 2021). We average these task accuracies to obtain the overall downstream score.

4.2 Main results

Results on Llama-3 8B and Qwen3-14B. We compare RTN (Banner et al., 2019), AWQ (Lin et al., 2024), GPTQ (Frantar et al., 2022), and GPTAQ (Li et al., 2025) before and after ReQuant at W4A16 and W4A4. Each pair in Table 1 holds bit-width, scales, zero-points, and grid fixed; ReQuant updates only integer codes.

Table 1: Average zero-shot accuracy (%) over ten tasks with QuaRot rotation. Each ReQuant row refines the initializer above it; the better result in each pair is bolded. L3 and Q3 denote Llama-3 and Qwen3. Full task scores appear in Appendix C (low-bit: Table 18); PPL/KL appear in Appendix B.

	L3-8B				Q3-14B		L3-70B
Method	W4A16	W4A4	W3A4	W2A4	W4A16	W4A4	W4A4
FP	66.19	66.19	66.19	66.19	71.24	71.24	72.92
RTN	61.78	53.33	36.41	30.18	68.03	64.97	32.78
RTN + ReQuant	**64.04**	**61.94**	**58.26**	**37.86**	**70.54**	**67.61**	**66.93**
AWQ	63.99	53.72	35.53	31.44	69.17	65.35	47.53
AWQ + ReQuant	**64.81**	**59.27**	**56.25**	**34.08**	**70.46**	**67.64**	**66.39**
GPTQ	65.22	61.29	57.10	35.88	70.52	67.95	58.09
GPTQ + ReQuant	**65.55**	**61.82**	**58.61**	**41.00**	**70.69**	**68.03**	**66.07**
GPTAQ	65.50	61.97	58.65	40.90	70.49	67.02	66.20
GPTAQ + ReQuant	**65.76**	**62.38**	**58.71**	**41.91**	**70.53**	**67.53**	**67.01**

Both precision settings use QuaRot (Ashkboos et al., 2024); Appendix B.6 reports a W4A16 ablation without rotation. Table 1 reports mean zero-shot accuracy over ten tasks, with individual scores in Appendix C. Appendices B.1 and B.2 give KL/PPL on WikiText-2, UltraChat-2k, and NuminaMath; Appendix B.4 reports runtime, and Appendix B.7 examines calibration-set size.

ReQuant increases mean accuracy in every reported pair for these two models and precision settings. The gains are generally larger for RTN and AWQ than for GPTQ and GPTAQ. For example, RTN gains 2.51 percentage points on Qwen3-14B at W4A16 and 8.61 points on Llama-3 8B at W4A4. These paired results show that revisiting integer assignments can substantially improve simple initializers without changing their quantization parameters.

The gains after stronger initialization are smaller. On Llama-3 8B at W4A16, mean accuracy rises from 65.22% to 65.55% for GPTQ and from 65.50% to 65.76% for GPTAQ. These results support further refinement after reconstruction-based PTQ, but do not imply uniform improvements across tasks or corpora; the detailed tables retain ties and regressions.

Refinement also narrows the accuracy gap between simple and stronger initializers. At W4A4, RTN+ReQuant nearly matches unrefined GPTAQ on Llama-3 8B (61.94% vs. 61.97%) and exceeds it on Qwen3-14B (67.61% vs. 67.02%). This comparison demonstrates competitive final accuracy from a simple starting point; the additional offline cost is reported separately.

Results on Llama-3 70B at W4A4. We further evaluate ReQuant on Llama-3 70B under the challenging W4A4 setting with QuaRot (Ashkboos et al., 2024) rotation. ReQuant is applied to the same four PTQ initializers: RTN, AWQ, GPTQ, and GPTAQ. Table 17 reports zero-shot accuracy across ten downstream tasks, and Table 8 in Appendix B.5 reports the corresponding KL divergence and perplexity.

At the 70B scale, ReQuant increases mean accuracy for all four initializers. GPTAQ improves from 66.20% to 67.01% ($+0.81$ percentage points), with lower PPL on all three corpora, although WikiText-2 KL increases. RTN+ReQuant reaches 66.93% mean accuracy, close to GPTAQ+ReQuant, and achieves lower KL/PPL on all three corpora. Thus, refinement yields competitive results from RTN at this scale, while the ranking of refined models depends on the evaluation metric.

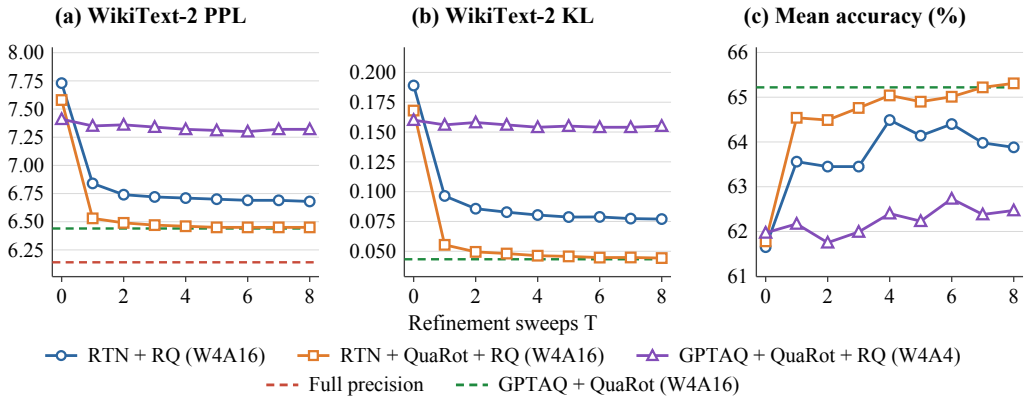


Figure 2: Refinement sweeps T versus WikiText-2 PPL/KL and mean zero-shot accuracy on Llama-3 8B. RQ denotes ReQuant; RTN+RQ uses no rotation, while the other two settings use QuaRot (Ashkboos et al., 2024). Dashed lines mark unrefined references. Lower PPL/KL and higher accuracy are better.

4.3 Lower-bit weight quantization

Table 1 reports paired W3A4/W2A4 results on Llama-3 8B with QuaRot (Ashkboos et al., 2024); Tables 18 and 6 retain all task scores and three-corpus KL/PPL. Mean accuracy rises in every pair. At W3A4, RTN and AWQ gain 21.85 and 20.72 percentage points, versus 0.06 for GPTAQ. At W2A4, GPTQ+ReQuant reaches 41.00%, slightly above unrefined GPTAQ (40.90%); refined RTN and AWQ remain below it. Gains depend on initialization and precision, and individual tasks can regress.

4.4 Refinement behavior and robustness

Figure 2 shows how refinement sweeps T affect PPL, KL, and mean accuracy on Llama-3 8B. RTN exhibits large early PPL/KL reductions, followed by smaller changes. Every accepted update decreases the empirical layer-wise objective, but held-out quality need not improve monotonically. Appendix C.1 gives settings and full results.

Cost–benefit of T. Table 19 records W4A16 quality and offline cost ($K{=}2$). Four sweeps take 81.80 minutes for RTN+QuaRot and 94.05 for GPTQ+QuaRot. The caption distinguishes reused main-table quality entries from the sweep measurements; deployment format remains unchanged.

Search sensitivity and seed stability. At fixed $T{=}4$ and seed 0, GPTQ+QuaRot+ReQuant yields WikiText PPL of 6.4266–6.4322 for $K{\in}\{1,2,3\}$. Three coordinate orders differ by at most 0.0076 in PPL and 0.37 percentage points in mean accuracy. Across five QuaRot seeds ($T{=}4$, $K{=}2$), WikiText PPL is $6.4277{\pm}0.0040$ and mean accuracy is $65.27{\pm}0.55\%$ (mean $\pm$ sample standard deviation). Appendix B.9 reports settings and all per-seed results. PPL varies little across these seeds, while task accuracy is more variable.

4.5 Comparison with FlexRound

Table 2 compares FlexRound (Lee et al., 2023) and ReQuant at W4A16 with the same $512{\times}2048$ WikiText-2 calibration budget, evaluation protocol, and RTX PRO 6000 timing hardware. Plain FlexRound does not use QuaRot; this *pipeline-level* comparison includes initialization and rotation effects.

The paired GPTQ+QuaRot control (Appendix B.8) fixes initialization and quantization parameters: ReQuant lowers WikiText PPL from 6.53 to 6.49 and raises mean accuracy from 65.22% to 65.55%, while NuminaMath PPL increases from 2.83 to 2.84.

Table 2: W4A16 pipeline comparison, timed on the same RTX PRO 6000. ReQuant uses $T{=}8$ for Llama-2 RTN and $T{=}4$ otherwise ($K{=}2$). Llama-3 ReQuant quality entries reuse Tables 1 and 4; hardware-matched timings are retained. W2/UC/NM denote WikiText-2, UltraChat-2k, and NuminaMath. Per-model column bests are bold.

		W2		UC		NM			
Model	Method	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	Acc.↑	Time (min)↓
Llama-2 7B	FlexRound	0.034	5.65	0.029	3.09	0.041	3.26	60.65	150.67
	RTN+QuaRot+ReQuant	0.020	5.59	0.019	3.06	0.021	3.22	60.69	95.72
	GPTQ+QuaRot+ReQuant	**0.016**	**5.57**	**0.016**	**3.05**	**0.018**	**3.20**	**60.96**	**70.00**
Llama-3 8B	FlexRound	0.075	6.69	0.067	3.73	0.081	2.97	65.08	179.50
	FlexRound+QuaRot	**0.048**	**6.47**	0.039	**3.58**	0.044	**2.84**	65.24	188.50
	RTN+QuaRot+ReQuant	0.079	6.71	0.063	3.70	0.068	2.91	64.04	**81.80**
	GPTQ+QuaRot+ReQuant	0.049	6.49	**0.036**	3.59	**0.041**	**2.84**	**65.55**	94.05

GPTQ+QuaRot+ReQuant improves all reported quality metrics over plain FlexRound on Llama-2 7B (Touvron et al., 2023b) and Llama-3 8B in less time. Against FlexRound+QuaRot on Llama-3 8B, it achieves higher mean accuracy (65.55%) and lower UltraChat/NuminaMath KL, but higher WikiText PPL/KL and UltraChat PPL; NuminaMath PPL ties.

4.6 Scalability to Qwen3-235B MoE

Table 3 evaluates Qwen3-235B-A22B (Yang et al., 2025) (235B total / 22B active parameters) at W4, using eight H200 GPUs, expert-parallel solving, QuaRot, and $512{\times}2048$ WikiText-2 calibration. Four ReQuant sweeps lower GPTQ+QuaRot's Top-20 KL on all three corpora and raise mean accuracy from 75.00% to 75.35%, but increase UltraChat/NuminaMath PPL and offline time; WikiText PPL ties at the reported precision. RTN+QuaRot+ReQuant takes 321.27 minutes at $T{=}4$, the shortest reported time. These results demonstrate offline feasibility at MoE scale, with metric-dependent gains.

Table 3: Qwen3-235B-A22B W4 scalability evaluation (8×H200). W2/UC/NM denote WikiText-2, UltraChat-2k, and NuminaMath; KL denotes Top-20 KL divergence. The best entry in each column is bolded.

		W2		UC		NM			
Method	T	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	Acc.↑	Time (min)↓
GPTQ+QuaRot	0	0.075	**4.53**	0.041	**3.98**	0.043	**2.49**	75.00	441.09
GPTQ+QuaRot+ReQuant	4	**0.064**	**4.53**	**0.035**	4.12	**0.040**	2.53	**75.35**	695.28
RTN+QuaRot+ReQuant	4	0.083	4.60	0.045	4.13	0.049	2.53	74.95	**321.27**
RTN+QuaRot+ReQuant	8	0.079	4.58	0.044	4.16	0.048	2.52	74.58	624.92

5 Conclusion

ReQuant refines completed PTQ outputs through loss-reducing integer-code updates on a fixed grid, preserving the deployment format and serving latency. Experiments across model families, bit-widths, and initializers, including Qwen3-235B MoE, show especially large gains for simple initializers and lower bit-widths. The sweep count T controls the offline budget.

Limitations. The coordinate-local search depends on the calibration set and inherited scales/zero-points. Offline cost grows with T, particularly relative to fast initializers such as RTN; fewer sweeps reduce this cost without changing inference. ReQuant optimizes a layer-wise calibration objective; we evaluate held-out performance empirically.

AI USE STATEMENT

Generative AI tools assisted with English-language editing, checking consistency across result tables, revising the interpretation of the corresponding comparisons, and preparing the LaTeX manuscript in the conference format. Numerical corrections followed source tables designated by the authors; this assistance did not produce new experimental measurements. The authors are responsible for the final content.

REPRODUCIBILITY STATEMENT

An anonymized code repository is available at `https://anonymous.4open.science/r/ReQuant-32133aad/`. Algorithm 1 specifies the refinement procedure, and Section 4.1 describes the model configurations, calibration data, and evaluation protocol. Appendix A derives and analyzes the updates; Appendix B.9 reports search-procedure ablations.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

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 (NeurIPS)*, 37, 2024.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen Technical Report. *arXiv preprint arXiv:2309.16609*, 2023.

Ron Banner, Yury Nahshan, and Daniel Soudry. Post-training 4-bit quantization of convolutional networks for rapid deployment. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. EfficientQAT: Efficient quantization-aware training for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 10081–10100, 2025.

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: Parameterized clipping activation for quantized neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. PaLM: Scaling language modeling with Pathways. *Journal of Machine Learning Research*, 24(240):1–113, 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, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. GPT3.int8(): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35: 30318–30332, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *Advances in Neural Information Processing Systems*, 36:10088–10115, 2023.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P. Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V. Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5547–5569. PMLR, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

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.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. In *Low-Power Computer Vision*, pp. 291–326. Chapman and Hall/CRC, 2022.

Ruihao Gong, Yifu Ding, Zining Wang, Chengtao Lv, Xingyu Zheng, Jinyang Du, Yang Yong, Shiqiao Gu, Haotong Qin, Jinyang Guo, et al. A survey of low-bit large language models: Basics, systems, and algorithms. *Neural Networks*, pp. 107856, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline Chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 62991–63010, 2023.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Advances in Neural Information Processing Systems*, 29, 2016.

Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Accurate post training quantization with small calibration sets. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4466–4475. PMLR, 2021.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2704–2713, 2018.

Wenjing Ke, Zhe Li, Dong Li, Lu Tian, and Emad Barsoum. DL-QAT: Weight-decomposed low-rank quantization-aware training for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 113–119, 2024.

Jung Hyun Lee, Jeonghoon Kim, Se Jung Kwon, and Dongsoo Lee. FlexRound: Learnable rounding based on element-wise division for post-training quantization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18913–18939. PMLR, 2023.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. NuminaMath: A large-scale math reasoning dataset. `https://huggingface.co/datasets/AI-MO/NuminaMath-CoT`, 2024.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations*, 2021.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. GPTAQ: Efficient finetuning-free quantization for asymmetric calibration. *arXiv preprint arXiv:2504.02692*, 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 (MLSys)*, 6:87–100, 2024.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, et al. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. LLM-QAT: Data-free quantization-aware training for large language models. In *Findings of the Association for Computational Linguistics: ACL*, pp. 467–484, 2024b.

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. *arXiv preprint arXiv:2405.16406*, 2024c.

Vladimir Malinovskii, Denis Mazur, Ivan Ilin, Denis Kuznedelev, Konstantin Burlachenko, Kai Yi, Dan Alistarh, and Peter Richtárik. PV-Tuning: Beyond straight-through estimation for extreme LLM compression. *Advances in Neural Information Processing Systems*, 37:5074–5121, 2024.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7197–7206. PMLR, 2020.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers*, pp. 1525–1534, 2016.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

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.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. SocialIQA: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

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

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

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 (ICML)*, pp. 38087–38099, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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.

A REQUANT DERIVATION DETAILS

A.1 REQUANT COORDINATE UPDATE DERIVATION

We derive ReQuant's row-wise quantities and explain how cached statistics score each candidate move. For one output row, let $\mathbf{w}$ be the full-precision row, $\mathbf{q}$ the current quantized row, and $\mathbf{e} := \mathbf{w} - \mathbf{q}$ the quantization error. With $\Delta\mathbf{X} := \widetilde{\mathbf{X}} - \mathbf{X}$, the activation-aware row loss in Eq. 5 can be rewritten as

$$\begin{aligned} L(\mathbf{e}) &= \|\mathbf{w}\mathbf{X} - \mathbf{q}\widetilde{\mathbf{X}}\|_2^2 \\ &= \|\mathbf{w}(\mathbf{X} - \widetilde{\mathbf{X}}) + (\mathbf{w} - \mathbf{q})\widetilde{\mathbf{X}}\|_2^2 \\ &= \|\mathbf{e}\widetilde{\mathbf{X}} - \mathbf{w}\Delta\mathbf{X}\|_2^2. \end{aligned}$$

Thus L is a quadratic function of $\mathbf{e}$. Defining

$$\widetilde{\mathbf{H}} := \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^\top, \qquad \mathbf{B} := \Delta\mathbf{X}\widetilde{\mathbf{X}}^\top,$$

and expanding the squared norm gives

$$L(\mathbf{e}) = \mathbf{e}\widetilde{\mathbf{H}}\mathbf{e}^\top - 2\mathbf{e}\mathbf{B}^\top\mathbf{w}^\top + \mathbf{w}\Delta\mathbf{X}\Delta\mathbf{X}^\top\mathbf{w}^\top.$$

The last term is constant with respect to $\mathbf{e}$, so the gradient vector used in the main text is

$$\mathbf{g} := \nabla_{\mathbf{e}} L(\mathbf{e}) = 2\big(\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B}\big).$$

Now consider changing coordinate j of the quantized row by an on-grid step Δq_j. Since $\mathbf{e} = \mathbf{w} - \mathbf{q}$, increasing q_j by Δq_j changes the error to

$$\mathbf{e}' = \mathbf{e} - \Delta q_j \mathbf{u}_j,$$

where $\mathbf{u}_j$ is the j-th unit vector. The exact loss change is

$$\begin{aligned} \Delta L(\Delta q_j) &:= L(\mathbf{e}') - L(\mathbf{e}) \\ &= \|(\mathbf{e} - \Delta q_j \mathbf{u}_j)\widetilde{\mathbf{X}} - \mathbf{w}\Delta\mathbf{X}\|_2^2 - \|\mathbf{e}\widetilde{\mathbf{X}} - \mathbf{w}\Delta\mathbf{X}\|_2^2 \\ &= -2\Delta q_j\, \mathbf{u}_j\widetilde{\mathbf{X}}(\mathbf{e}\widetilde{\mathbf{X}} - \mathbf{w}\Delta\mathbf{X})^\top + (\Delta q_j)^2\|\mathbf{u}_j\widetilde{\mathbf{X}}\|_2^2 \\ &= -2\Delta q_j\, (\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B})_j + (\Delta q_j)^2\widetilde{H}_{jj} \\ &= -\Delta q_j\, g_j + (\Delta q_j)^2\widetilde{H}_{jj}. \end{aligned}$$

This proves Eq. 7. We evaluate a candidate move using only g_j and $\widetilde{H}_{jj}$, without recomputing the row residual.

The integer quantization code determines the feasible candidates. If $q_j = (z_j - o_j)s_j$, any valid move has the form $\Delta q_j = ks_j$, where $k \in \mathbb{Z} \setminus \{0\}$ and $z_{\min} \le z_j + k \le z_{\max}$. Finite enumeration in Eq. 9 selects the best coordinate move. After accepting a move, ReQuant updates its state exactly. The error update is immediate from $\mathbf{e}' = \mathbf{e} - \Delta q_j^\star \mathbf{u}_j$, and the gradient vector satisfies

$$\begin{aligned} \mathbf{g}' &= 2\big(\mathbf{e}'\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B}\big) \\ &= 2\big((\mathbf{e} - \Delta q_j^\star \mathbf{u}_j)\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B}\big) \\ &= \mathbf{g} - 2\Delta q_j^\star \widetilde{\mathbf{H}}_{j,:}. \end{aligned}$$

Finally, Eq. 7 gives the exact loss difference, so ReQuant updates the scalar loss by $L \leftarrow L + \Delta L(\Delta q_j^\star)$. Together these identities justify the incremental updates used in Algorithm 1.

A.2 FINITE TERMINATION OF REQUANT

We prove finite termination for one row under fixed calibration statistics and a fixed quantization grid; the layer-wise result follows from the row decomposition in Eq. 4. Let the feasible values of coordinate j be the finite set $\mathcal{Q}_j$. Under b-bit quantization, $|\mathcal{Q}_j| \le 2^b$. The feasible set for one quantized row is therefore the Cartesian product

$$\mathcal{Q} = \mathcal{Q}_1 \times \mathcal{Q}_2 \times \cdots \times \mathcal{Q}_{d_{\text{col}}},$$

whose cardinality satisfies

$$|\mathcal{Q}| = \prod_{j=1}^{d_{\text{col}}} |\mathcal{Q}_j| \leq (2^b)^{d_{\text{col}}} = 2^{bd_{\text{col}}}.$$

Hence $\mathcal{Q}$ is finite.

Let $\mathbf{q}^{(r)} \in \mathcal{Q}$ denote the quantized row after the r-th accepted update, and let

$$\mathbf{e}^{(r)} = \mathbf{w} - \mathbf{q}^{(r)}$$

be the corresponding quantization error. ReQuant accepts an update only if

$$\Delta L^{(r)} := L(\mathbf{e}^{(r+1)}) - L(\mathbf{e}^{(r)}) < 0.$$

Therefore every accepted update strictly decreases the row objective:

$$L(\mathbf{e}^{(0)}) > L(\mathbf{e}^{(1)}) > L(\mathbf{e}^{(2)}) > \cdots .$$

We now prove by contradiction that infinitely many accepted updates are impossible. Suppose ReQuant accepted infinitely many updates. Then it would generate an infinite sequence

$$\mathbf{q}^{(0)}, \mathbf{q}^{(1)}, \mathbf{q}^{(2)}, \ldots$$

with every $\mathbf{q}^{(r)} \in \mathcal{Q}$. Since $\mathcal{Q}$ is finite, any infinite sequence in $\mathcal{Q}$ must repeat an element. Hence there exist two indices $a < b$ such that

$$\mathbf{q}^{(a)} = \mathbf{q}^{(b)}.$$

Because $\mathbf{e} = \mathbf{w} - \mathbf{q}$, this equality also implies

$$\mathbf{e}^{(a)} = \mathbf{w} - \mathbf{q}^{(a)} = \mathbf{w} - \mathbf{q}^{(b)} = \mathbf{e}^{(b)},$$

and thus

$$L(\mathbf{e}^{(a)}) = L(\mathbf{e}^{(b)}).$$

However, the sequence contains at least one accepted update between a and b, and every accepted update strictly decreases L. Hence

$$L(\mathbf{e}^{(b)}) < L(\mathbf{e}^{(a)}),$$

which contradicts $L(\mathbf{e}^{(a)}) = L(\mathbf{e}^{(b)})$.

Thus ReQuant cannot accept infinitely many updates. Since each accepted update visits a new quantized row, the number of accepted updates is at most

$$|\mathcal{Q}| - 1 \leq 2^{bd_{\text{col}}} - 1.$$

This bound is extremely loose, but it establishes finite termination under fixed calibration statistics.

A.3 Complexity Analysis

We analyze the cost of ReQuant for one linear layer with $\mathbf{W} \in \mathbb{R}^{d_{\text{row}} \times d_{\text{col}}}$ and $\mathbf{X}, \widetilde{\mathbf{X}} \in \mathbb{R}^{d_{\text{col}} \times m}$, where m is the number of calibration tokens. As in Section 3.2, K denotes the integer-offset search radius in each direction. Each coordinate has at most $\min(2K, 2^b - 1)$ feasible nonzero moves under b-bit quantization, with fewer near code boundaries. Thus candidate enumeration costs $O(K)$; the factor of two does not change the asymptotic bounds below.

Naive candidate evaluation. To test whether changing one entry $W_{q,i,j}$ improves the objective, the naive approach recomputes the full layer reconstruction loss $\|\mathbf{W}\mathbf{X} - \mathbf{W}_q\widetilde{\mathbf{X}}\|_F^2$. This requires a matrix multiplication and costs $O(md_{\text{row}}d_{\text{col}})$ per candidate. Since a layer contains $d_{\text{row}}d_{\text{col}}$ coordinates and each coordinate has at most $2K$ candidates, naive exhaustive evaluation would cost

$$O(K\, m\, d_{\text{row}}^2 d_{\text{col}}^2),$$

which is prohibitive for LLM layers.

Effect of row-wise decomposition. By Eq. 4, changing $W_{q,i,j}$ affects only the i-th row loss

$$L_i = \|\mathbf{W}_{i,:}\mathbf{X} - \mathbf{W}_{q,i,:}\widetilde{\mathbf{X}}\|_2^2.$$

If this row loss were recomputed from scratch for each candidate, the cost would be $O(md_{\text{col}})$ per candidate, reducing the full-layer factor d_{row} and enabling all rows to be processed independently. This is still expensive because it repeatedly accesses the calibration activations.

Closed-form candidate scoring. ReQuant instead uses the quadratic form in Eq. 5. Once $\mathbf{g} = 2(\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B})$ is available, Eq. 7 scores one candidate using only Δq_j, g_j, and $\widetilde{H}_{jj}$, so each candidate costs $O(1)$. Scanning at most $2K$ candidates at one coordinate costs $O(K)$; scanning all coordinates of all rows in one sweep costs

$$O(d_{\text{row}}d_{\text{col}}K).$$

Accepted-update maintenance. When a move is accepted, ReQuant refreshes the gradient vector by $\mathbf{g} \leftarrow \mathbf{g} - 2\Delta q_j^\star \widetilde{\mathbf{H}}_{j,:}$. This costs $O(d_{\text{col}})$, since $\widetilde{\mathbf{H}}_{j,:}$ has length d_{col}. Recomputing $\mathbf{g} = 2(\mathbf{e}\widetilde{\mathbf{H}} - \mathbf{w}\mathbf{B})$ from scratch for one row would cost $O(d_{\text{col}}^2)$. If A moves are accepted in one sweep, the state-maintenance cost is therefore $O(Ad_{\text{col}})$. In the worst case, $A \le d_{\text{row}}d_{\text{col}}$, giving $O(d_{\text{row}}d_{\text{col}}^2)$ per sweep for accepted-update maintenance.

Precomputation and initialization. Before refinement, ReQuant computes

$$\widetilde{\mathbf{H}} = \widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^\top, \qquad \mathbf{B} = \Delta\mathbf{X}\widetilde{\mathbf{X}}^\top,$$

which costs $O(md_{\text{col}}^2)$ and stores $O(d_{\text{col}}^2)$ statistics shared by all output rows. Initializing the gradient matrix for all rows,

$$\mathbf{G} = 2(\mathbf{E}\widetilde{\mathbf{H}} - \mathbf{W}\mathbf{B}), \qquad \mathbf{E} = \mathbf{W} - \mathbf{Q},$$

costs $O(d_{\text{row}}d_{\text{col}}^2)$. Initializing all row losses from residuals costs $O(md_{\text{row}}d_{\text{col}})$, which is usually dominated by the matrix-statistic and gradient-initialization terms.

Total cost. For T refinement sweeps and average accepted updates A_{avg} per sweep, the layer-wise ReQuant complexity is

$$O(md_{\text{col}}^2) + O(d_{\text{row}}d_{\text{col}}^2) + O(Td_{\text{row}}d_{\text{col}}K) + O(TA_{\text{avg}}d_{\text{col}}).$$

Using $A_{\text{avg}} \le d_{\text{row}}d_{\text{col}}$, the worst-case bound becomes

$$O(md_{\text{col}}^2) + O((T+1)d_{\text{row}}d_{\text{col}}^2) + O(Td_{\text{row}}d_{\text{col}}K).$$

With a small search radius $K \ll d_{\text{col}}$, the dominant terms are

$$O(md_{\text{col}}^2 + (T+1)d_{\text{row}}d_{\text{col}}^2).$$

B ADDITIONAL EXPERIMENTAL RESULTS

This appendix collects supplementary experimental results that did not fit in the main text. All *KL/PPL* tables below are evaluated on WikiText-2, UltraChat-2k, and NuminaMath; we abbreviate the three column groups as **W2**, **UC**, and **NM**, matching the headers. **KL** is the (unscaled) KL divergence to the full-precision model on each evaluation text. **PPL** is standard *token-level* perplexity on the same text. Unless noted otherwise, we report PPL to two decimal places and KL to three. Lower is better for both. Unless noted otherwise, **within each baseline pair** (each PTQ initializer vs. the same initializer after ReQuant), **the better entry is bolded.** Section B.1 reports these metrics for Llama-3 8B and Qwen3-14B at W4A16. Section B.2 reports them at W4A4 for the same models. Section B.5 reports them on Llama-3 70B at W4A4. Section B.6 ablates QuaRot by re-running the W4A16 experiments without any orthogonal preprocessing. Section B.4 reports the runtime overhead on Qwen3-14B. Section B.3 reports KL/PPL for the W3A4/W2A4 setting in Section 4.3. Section B.7 plots WikiText-2 PPL/KL and average downstream accuracy as a function of calibration sample size for GPTAQ with QuaRot and ReQuant at W4A4 on Llama-3 8B. Sections B.8–B.9 add FlexRound/QuaRot controls and search-procedure ablations (K, coordinate order, seed). The Qwen3-235B MoE scalability study is reported in Section 4.6.

B.1 W4A16 RESULTS ON LLAMA-3 8B AND QWEN3-14B

At W4A16 with QuaRot, Table 4 shows broad KL/PPL reductions after RTN and AWQ initialization on Llama-3 8B and Qwen3-14B. GPTAQ also improves on most metrics, with ties at the reported precision. GPTQ yields smaller, mixed changes, including higher NuminaMath PPL on both models. Thus, refinement benefits depend on the initializer and corpus.

Table 4: KL and PPL under W4A16 quantization on Llama-3 8B and Qwen3-14B with QuaRot (Ashkboos et al., 2024) rotation.

	Llama-3 8B						Qwen3-14B					
	W2		UC		NM		W2		UC		NM	
Method	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓
FP	–	6.14	–	3.47	–	2.71	–	8.65	–	4.98	–	3.36
RTN	0.174	7.63	0.116	3.94	0.100	3.05	0.176	10.22	0.145	5.20	0.102	**3.27**
RTN + ReQuant	**0.079**	**6.71**	**0.063**	**3.70**	**0.068**	**2.91**	**0.031**	**8.85**	**0.036**	**4.98**	**0.029**	3.33
AWQ	0.091	6.82	0.064	3.69	0.054	**2.87**	0.140	9.68	0.117	5.16	0.078	3.43
AWQ + ReQuant	**0.069**	**6.65**	**0.051**	**3.65**	**0.053**	**2.87**	**0.052**	**8.96**	**0.054**	**5.04**	**0.036**	**3.39**
GPTQ	0.054	6.53	0.038	**3.59**	**0.041**	**2.83**	0.027	**8.78**	**0.031**	4.97	**0.024**	**3.34**
GPTQ + ReQuant	**0.049**	**6.49**	**0.036**	**3.59**	**0.041**	2.84	**0.025**	8.81	**0.031**	**4.94**	0.025	3.36
GPTAQ	0.043	6.44	0.032	3.59	0.040	2.84	0.027	8.83	0.033	4.96	**0.026**	3.36
GPTAQ + ReQuant	**0.040**	**6.41**	**0.031**	**3.58**	**0.028**	**2.83**	**0.026**	**8.80**	**0.033**	**4.91**	**0.026**	**3.34**

B.2 W4A4 RESULTS ON LLAMA-3 8B AND QWEN3-14B

At W4A4, Table 5 shows larger WikiText-2 KL/PPL than the corresponding W4A16 configurations. ReQuant reduces WikiText-2 KL/PPL for all initializer–model pairs, with particularly large reductions for RTN and AWQ on Llama-3 8B. Results on other corpora are mixed: Qwen3-14B NuminaMath PPL rises after RTN, AWQ, and GPTQ refinement. These metric-specific outcomes complement the mean-accuracy gains in Table 1.

B.3 KL/PPL UNDER W3A4 AND W2A4 ON LLAMA-3 8B

Table 6 reports WikiText-2, UltraChat-2k, and NuminaMath KL divergence and perplexity for the low-bit setting in Section 4.3. The corresponding zero-shot accuracies appear in Table 18.

Table 5: KL and PPL under W4A4 quantization on Llama-3 8B and Qwen3-14B with QuaRot (Ashkboos et al., 2024) rotation.

	Llama-3 8B						Qwen3-14B					
	W2		UC		NM		W2		UC		NM	
Method	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓
FP	–	6.14	–	3.47	–	2.71	–	8.65	–	4.98	–	3.36
RTN	0.538	13.21	0.401	5.71	0.304	3.89	0.322	11.44	0.255	5.37	0.186	**3.41**
RTN + ReQuant	**0.165**	**7.38**	**0.130**	**3.97**	**0.122**	**3.10**	**0.149**	**9.69**	**0.138**	**5.10**	**0.099**	3.46
AWQ	0.408	10.99	0.312	5.08	0.220	2.96	0.271	10.43	0.224	5.24	0.153	**3.42**
AWQ + ReQuant	**0.221**	**7.94**	**0.166**	**4.14**	**0.115**	**2.56**	**0.165**	**9.80**	**0.150**	**5.12**	**0.101**	3.48
GPTQ	0.190	7.86	0.143	4.07	0.121	3.12	0.155	9.71	0.131	**5.03**	**0.088**	**3.37**
GPTQ + ReQuant	**0.167**	**7.46**	**0.122**	**3.95**	**0.113**	**3.08**	**0.143**	**9.61**	**0.128**	5.06	0.089	3.43
GPTAQ	0.162	7.40	0.115	3.93	**0.110**	**3.06**	0.158	9.73	0.149	**5.03**	0.106	3.47
GPTAQ + ReQuant	**0.158**	**7.36**	**0.113**	**3.91**	0.112	3.07	**0.152**	**9.67**	**0.145**	5.06	**0.101**	**3.42**

Table 6: KL divergence and perplexity (PPL) on Llama-3 8B at W3A4 and W2A4 with QuaRot (Ashkboos et al., 2024) rotation, evaluated on WikiText-2 (W2), UltraChat-2k (UC), and NuminaMath (NM). All four initializers are reported; the better value in each initializer/refinement pair is bolded.

		W2		UC		NM	
Precision	Method	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓
W3A4	RTN	1.690	87.59	1.200	20.81	0.926	9.31
	RTN + ReQuant	**0.266**	**8.38**	**0.246**	**4.50**	**0.241**	**3.53**
	AWQ	1.490	58.87	1.150	18.91	0.854	8.77
	AWQ + ReQuant	**0.330**	**9.14**	**0.295**	**4.80**	**0.289**	**3.76**
	GPTQ	0.305	9.21	0.269	4.74	0.245	3.60
	GPTQ + ReQuant	**0.251**	**8.19**	**0.220**	**4.38**	**0.216**	**3.47**
	GPTAQ	0.253	8.27	**0.212**	4.35	0.211	3.42
	GPTAQ + ReQuant	**0.242**	**8.13**	0.213	4.35	**0.207**	**3.41**
W2A4	RTN	4.000	34060.87	3.810	27544.27	4.240	26438.09
	RTN + ReQuant	**0.994**	**20.55**	**1.160**	**15.40**	**1.470**	**17.96**
	AWQ	3.730	115110.90	4.160	170269.44	4.460	164779.34
	AWQ + ReQuant	**1.430**	**38.59**	**1.810**	**41.91**	**2.470**	**97.71**
	GPTQ	1.090	31.98	1.180	21.57	1.100	12.82
	GPTQ + ReQuant	**0.735**	**15.06**	**0.838**	**9.71**	**0.995**	**9.10**
	GPTAQ	0.743	15.49	0.811	9.57	1.050	10.18
	GPTAQ + ReQuant	**0.701**	**14.38**	**0.789**	**9.15**	**1.040**	**9.81**

B.4 Runtime results on Qwen3-14B

Table 7 reports the end-to-end quantization time on Qwen3-14B for different PTQ initializers with QuaRot, both with and without ReQuant. For example, RTN+QuaRot uses 15.6 minutes and RTN+QuaRot+ReQuant uses 117.1 minutes; GPTAQ+QuaRot uses 35.3 minutes and GPTAQ+QuaRot+ReQuant uses 128.6 minutes. T controls the offline budget (Section 4.4).

This is an offline preparation cost: refinement preserves the quantized representation and introduces no additional inference operators. Avoiding backpropagation and optimizer states does not eliminate refinement overhead; ReQuant still computes and caches activation statistics and update scores.

Table 7: Runtime on Qwen3-14B.

Method	Runtime
RTN + QuaRot	15.6 min
RTN + QuaRot + ReQuant	117.1 min
AWQ + QuaRot	23.9 min
AWQ + QuaRot + ReQuant	109.8 min
GPTQ + QuaRot	31.4 min
GPTQ + QuaRot + ReQuant	128.6 min
GPTAQ + QuaRot	35.3 min
GPTAQ + QuaRot + ReQuant	128.6 min

B.5 KL DIVERGENCE AND PERPLEXITY RESULTS ON LLAMA-3 70B

Table 8 focuses on Llama-3 70B at W4A4 under QuaRot (Ashkboos et al., 2024). ReQuant reduces all six KL/PPL metrics for RTN, AWQ, and GPTQ, with the largest absolute reductions for RTN. For GPTAQ, PPL decreases on all three corpora, while WikiText-2 KL rises from 0.693 to 0.707. The latter pair illustrates that closer agreement with the full-precision distribution and lower perplexity need not coincide.

Table 8: KL and PPL under W4A4 quantization on Llama-3 70B with QuaRot (Ashkboos et al., 2024) rotation.

	W2		UC		NM	
Method	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓
FP	–	2.86	–	3.07	–	2.40
RTN	1.953	60.14	1.405	27.40	1.241	13.34
RTN + ReQuant	**0.678**	**6.00**	**0.126**	**3.49**	**0.143**	**2.76**
AWQ	1.158	14.26	0.570	6.33	0.550	4.58
AWQ + ReQuant	**0.683**	**6.07**	**0.145**	**3.56**	**0.160**	**2.80**
GPTQ	0.753	7.63	0.307	4.41	0.269	3.21
GPTQ + ReQuant	**0.743**	**6.59**	**0.168**	**3.65**	**0.182**	**2.87**
GPTAQ	**0.693**	6.33	0.130	3.51	0.154	2.80
GPTAQ + ReQuant	0.707	**6.27**	**0.127**	**3.50**	**0.148**	**2.78**

B.6 W4A16 RESULTS WITHOUT QUAROT ROTATION

Tables 9 and 10 repeat the Llama-3 8B W4A16 study *without* QuaRot orthogonal preprocessing. ReQuant generally lowers KL/PPL relative to the corresponding initializer, with some metric-specific regressions. Mean accuracy increases for RTN ($61.65 \rightarrow 63.20$, $+1.55$ percentage points), AWQ ($63.50 \rightarrow 64.06$), and GPTAQ ($64.28 \rightarrow 64.68$), but decreases for GPTQ ($64.63 \rightarrow 64.41$) despite lower WikiText-2 KL/PPL. Refinement therefore remains useful without rotation, but does not improve mean accuracy for every initializer in this ablation.

Table 9: KL and PPL under W4A16 on Llama-3 8B *without* QuaRot rotation.

	W2		UC		NM	
Method	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓
FP	–	6.14	–	3.47	–	2.71
RTN	0.255	8.53	0.218	4.42	0.190	3.37
RTN + ReQuant	**0.103**	**6.91**	**0.091**	**3.81**	**0.131**	**3.15**
AWQ	0.115	7.06	0.080	3.77	**0.079**	**2.96**
AWQ + ReQuant	**0.082**	**6.75**	**0.070**	**3.72**	0.083	2.97
GPTQ	0.090	6.81	**0.092**	**3.82**	**0.100**	**3.03**
GPTQ + ReQuant	**0.078**	**6.68**	0.101	3.86	0.105	3.04
GPTAQ	0.073	**6.65**	**0.116**	**3.94**	**0.109**	**3.05**
GPTAQ + ReQuant	**0.073**	**6.65**	0.123	3.97	0.114	3.07

Table 10: Zero-shot accuracy (%) under W4A16 on Llama-3 8B *without* QuaRot rotation. The two panels cover the same runs; Avg. is the ten-task mean.

Method	ARC-C	ARC-E	BoolQ	CEval	HellaSwag	Avg.
FP	53.58	77.69	81.25	48.44	79.17	66.19
RTN	48.98	75.59	73.09	38.04	75.36	61.65
RTN + ReQuant	**50.68**	**76.14**	**77.65**	**40.86**	**76.40**	**63.20**
AWQ	**50.85**	73.82	78.53	43.24	**77.60**	63.50
AWQ + ReQuant	48.89	**75.34**	**80.58**	**43.98**	77.49	**64.06**
GPTQ	49.06	73.82	**81.71**	**44.58**	77.12	**64.63**
GPTQ + ReQuant	**51.45**	**76.18**	78.81	41.16	**77.53**	64.41
GPTAQ	50.43	74.75	79.36	42.50	**77.81**	64.28
GPTAQ + ReQuant	**50.51**	**76.56**	**80.46**	**44.21**	77.45	**64.68**

Method	LAMBADA	OBQA	PIQA	SIQA	Winogrande	Avg.
FP	75.70	45.0	80.90	46.88	73.24	66.19
RTN	70.00	42.0	77.91	**44.37**	71.11	61.65
RTN + ReQuant	**72.25**	**43.2**	**78.62**	43.96	**72.22**	**63.20**
AWQ	70.29	43.0	**79.71**	46.11	71.90	63.50
AWQ + ReQuant	**71.24**	**44.2**	79.54	**46.21**	**73.16**	**64.06**
GPTQ	**72.91**	45.6	**80.14**	**46.78**	74.59	**64.63**
GPTQ + ReQuant	72.13	**45.8**	79.54	46.42	**75.30**	64.41
GPTAQ	**71.82**	45.2	**79.49**	**46.21**	**75.22**	64.28
GPTAQ + ReQuant	71.78	**45.8**	79.05	46.06	74.90	**64.68**

B.7 Effect of calibration sample size on Llama-3 8B at W4A4 (GPTAQ + QuaRot + ReQuant)

Figure 3 studies calibration-set size at **W4A4** on Llama-3 8B, with other settings following Section 4.1. We report WikiText-2 PPL/KL and mean zero-shot accuracy from lm-eval-harness for GPTAQ (Li et al., 2025) with QuaRot (Ashkboos et al., 2024), before and after ReQuant. PPL/KL generally decrease as the calibration set grows, with smaller changes at larger sample counts; mean accuracy fluctuates rather than improving monotonically.

At each plotted sample count, ReQuant yields lower PPL/KL and higher mean accuracy than GPTAQ+QuaRot alone. This paired pattern supports refinement across the tested calibration budgets, without implying that more calibration data necessarily improves every downstream metric.

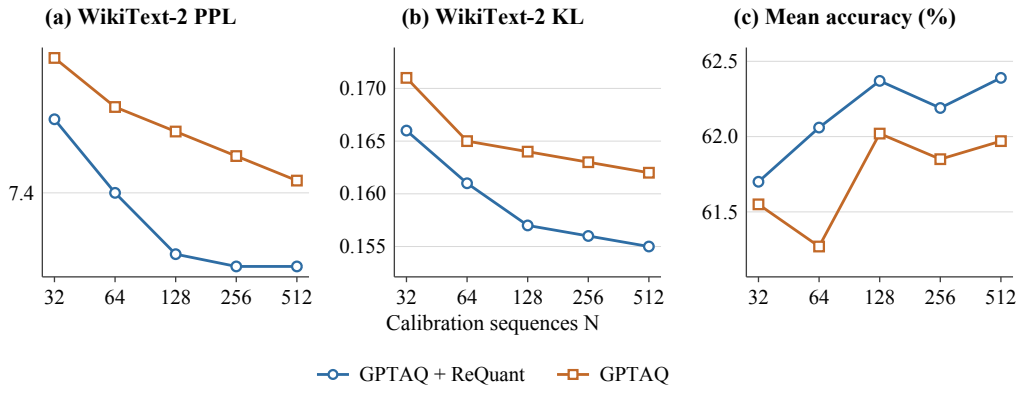


Figure 3: W4A4: sensitivity to calibration sample size for GPTAQ + QuaRot + ReQuant on Llama-3 8B. (left) WikiText-2 perplexity, (center) WikiText-2 KL divergence to the FP model, (right) average zero-shot accuracy (ten-task mean). Lower is better for PPL and KL; higher is better for accuracy.

B.8 Separating QuaRot from FlexRound comparisons

Table 11 reports controlled add-ons around the FlexRound comparison in Section 4.5 on Llama-3 8B under W4A16. Adding QuaRot to FlexRound improves all reported PPL/KL metrics and mean accuracy in this setting, showing that rotation contributes to the pipeline-level comparison. Pairing QuaRot with ReQuant improves all six PPL/KL metrics of the RTN+ReQuant pipeline and raises average accuracy from 63.20% to 64.04%. The paired GPTQ+QuaRot vs. GPTQ+QuaRot+ReQuant comparison isolates ReQuant under identical initialization, grid, scales, and zero-points. At the reported precision, three of six PPL/KL metrics improve, two remain unchanged, and NuminaMath PPL increases from 2.83 to 2.84; average accuracy rises from 65.22% to 65.55% (+0.33 percentage points).

Table 11: Controlled W4A16 comparisons on Llama-3 8B that separate QuaRot from ReQuant. The RTN+ReQuant row without QuaRot reuses Tables 9 and 10; RTN/GPTQ rows with QuaRot reuse Tables 4 and 1. The best entry in each column is bolded.

Method	Wiki PPL↓	Wiki KL↓	UC PPL↓	UC KL↓	NM PPL↓	NM KL↓	Acc.↑
FlexRound	6.69	0.075	3.73	0.067	2.97	0.081	65.08
FlexRound+QuaRot	**6.47**	**0.048**	**3.58**	0.039	2.84	0.044	65.24
RTN+ReQuant (T=4, no QuaRot)	6.91	0.103	3.81	0.091	3.15	0.131	63.20
RTN+QuaRot+ReQuant (T=4)	6.71	0.079	3.70	0.063	2.91	0.068	64.04
GPTQ+QuaRot (T=0)	6.53	0.054	3.59	0.038	**2.83**	**0.041**	65.22
GPTQ+QuaRot+ReQuant (T=4)	6.49	0.049	3.59	**0.036**	2.84	**0.041**	**65.55**

B.9 SEARCH SENSITIVITY AND SEED STABILITY

Unless noted, experiments below use Llama-3 8B, W4A16, GPTQ+QuaRot+ReQuant, $T{=}4$, and seed 0.

Search radius K. Candidate integer offsets are $\{\pm1, \ldots, \pm K\}$. Table 12 shows similar PPL/KL and runtime for $K{\in}\{1, 2, 3\}$, with more variation in mean accuracy. At the displayed precision, GPTQ PPL ties across all three radii, and $K{=}2, 3$ tie on KL. For RTN, $K{=}2$ yields the highest mean accuracy, while $K{=}1$ has the lowest KL. These results support $K{=}2$ as a practical default, not a uniformly optimal radius. For RTN at $K{=}2$, the PPL/KL entries come from a separate run with the same nominal settings because the K-sweep PPL/KL evaluation was flagged as anomalous; accuracy comes from the K-sweep. These entries therefore do not constitute a single matched evaluation.

Table 12: Search-radius ablation at fixed $T{=}4$ on Llama-3 8B (W4A16).

Initializer	K	Wiki PPL↓	Wiki KL↓	Acc.↑	Time (min)↓
RTN+QuaRot	1	**6.52**	**0.054**	64.84	80.72
	2	**6.52**	0.055	**65.48**	80.89
	3	6.53	0.055	65.08	**80.19**
GPTQ+QuaRot	1	**6.43**	0.042	65.41	**92.93**
	2	**6.43**	**0.041**	65.33	96.36
	3	**6.43**	**0.041**	**65.73**	95.61

Coordinate ordering. Table 13 compares forward ($j{=}1 \ldots d_{\text{col}}$), reverse, and a random-fixed within-row permutation reused across sweeps. WikiText PPL differs by at most 0.0076 and mean accuracy by 0.37 percentage points. The tested orders yield similar PPL/KL but different task means; random-fixed takes longer in this implementation. These results indicate limited sensitivity to ordering in this configuration, rather than order-independent outcomes. Its reported time excludes a recorded pause of approximately 78.91 minutes between layers 27 and 28; forward and reverse use their recorded run times.

Table 13: Within-row coordinate-order ablation for GPTQ+QuaRot+ReQuant ($T{=}4$, $K{=}2$) on Llama-3 8B.

Order	Wiki PPL↓	Wiki KL↓	Acc.↑	Time (min)↓
Forward	**6.43**	**0.041**	65.33	96.36
Reverse	**6.43**	0.042	**65.70**	**96.06**
Random-fixed	**6.43**	0.042	65.60	122.65

Accepted-update statistics. Reverse and random-fixed accept $220{,}663{,}451$ and $211{,}482{,}930$ of $27{,}917{,}287{,}424$ examined row-coordinate visits, respectively (0.7904% and 0.7575%). These aggregates describe update frequency over all four sweeps, not a per-sweep convergence curve; forward counts were not recorded in this summary.

QuaRot seed stability. We vary the QuaRot seed over $\{0, 1, 2, 3, 4\}$ for Llama-3 8B with GPTQ+QuaRot+ReQuant at W4A16, holding the calibration data (512×2048 WikiText-2 sequences), $T=4$, and $K=2$ fixed. Table 14 retains every reported seed and task. WikiText PPL varies little (6.4277 ± 0.0040), while mean accuracy ranges from 64.68% to 66.09%. C-Eval has the largest task-level standard deviation (3.00 percentage points), so the aggregate should not be read as uniform stability across tasks.

This is a separate seed-sweep experiment: its means do not replace the main-table single-run results. The experiment measures variation in the refined pipeline, rather than paired gains over seed-matched unrefined models or sensitivity to calibration-sample selection.

Table 14: QuaRot-seed sensitivity of GPTQ+QuaRot+ReQuant on Llama-3 8B at W4A16 ($T=4$, $K=2$), with fixed calibration data. Panel (a) reports three-corpus quality, ten-task mean accuracy, and offline quantization time; panel (b) retains all task accuracies (%). Summary statistics use the five reported values and sample standard deviation ($n-1$ denominator), not standard error. W2/UC/NM denote WikiText-2, UltraChat-2k, and NuminaMath.

(a) Quality and offline cost

Metric	Seed 0	Seed 1	Seed 2	Seed 3	Seed 4	Mean ± Std
W2 PPL	6.4266	6.4248	6.4278	6.4249	6.4345	6.4277 ± 0.0040
W2 KL	0.0412	0.0414	0.0409	0.0414	0.0420	0.0414 ± 0.0004
UC PPL	3.5835	3.5781	3.5879	3.5872	3.5797	3.5833 ± 0.0044
UC KL	0.0326	0.0317	0.0329	0.0322	0.0327	0.0324 ± 0.0005
NM PPL	2.8314	2.8353	2.8194	2.8388	2.8165	2.8283 ± 0.0098
NM KL	0.0424	0.0400	0.0350	0.0395	0.0358	0.0385 ± 0.0031
Avg. acc. (%)	65.33	64.68	66.09	64.84	65.42	65.27 ± 0.55
Time (s)	5671.16	5755.62	5637.91	5696.87	5698.94	5692.10 ± 43.24
Time (min)	94.52	95.93	93.97	94.95	94.98	94.87 ± 0.72

(b) Downstream accuracy (%)

Task	Seed 0	Seed 1	Seed 2	Seed 3	Seed 4	Mean ± Std
ARC-C	53.16	52.73	53.84	52.99	53.16	53.18 ± 0.41
ARC-E	77.27	76.64	79.00	77.27	76.26	77.29 ± 1.05
BoolQ	80.86	80.95	81.56	81.53	81.96	81.37 ± 0.46
C-Eval	45.91	40.71	48.22	43.91	47.33	45.22 ± 3.00
HellaSwag	78.38	77.98	78.56	77.78	77.51	78.04 ± 0.43
LAMBADA	74.02	75.32	75.24	74.29	74.58	74.69 ± 0.57
OBQA	44.40	43.80	44.20	42.60	43.40	43.68 ± 0.72
PIQA	79.76	80.03	80.09	79.98	79.60	79.89 ± 0.21
Winogrande	73.01	72.38	73.80	72.22	73.64	73.01 ± 0.71
SocialIQA	46.52	46.26	46.42	45.85	46.78	46.37 ± 0.34

C Detailed task scores and refinement sweeps

This appendix retains the individual task scores summarized in Table 1. The split tables below abbreviate ReQuant as RQ.

Table 15: Zero-shot accuracy (%) on Llama-3 8B with QuaRot at W4A16 and W4A4. The two panels report complementary sets of tasks for the same runs. Avg. is the ten-task mean.

Precision	Method	ARC-C	ARC-E	BoolQ	CEval	HellaSwag	Avg.
FP	–	53.58	77.69	81.25	48.44	79.17	66.19
W4A16	RTN	47.53	69.74	78.01	36.63	75.78	61.78
	RTN + RQ	**50.68**	**75.93**	**80.18**	**41.98**	**77.06**	**64.04**
	AWQ	50.34	**77.53**	79.05	38.86	77.34	63.99
	AWQ + RQ	**51.79**	75.59	**82.11**	**44.65**	**77.72**	**64.81**
	GPTQ	51.88	77.65	80.76	45.32	**78.13**	65.22
	GPTQ + RQ	**54.18**	**77.69**	**82.14**	**45.47**	78.01	**65.55**
	GPTAQ	**53.33**	77.95	**81.80**	44.73	77.97	65.50
	GPTAQ + RQ	**53.33**	**78.16**	80.86	**47.47**	**78.17**	**65.76**
W4A4	RTN	39.85	62.58	67.40	31.05	66.16	53.33
	RTN + RQ	**46.25**	**74.58**	**78.07**	**38.11**	**74.46**	**61.94**
	AWQ	37.97	60.44	65.93	28.97	68.89	53.72
	AWQ + RQ	**44.28**	**69.95**	**73.67**	**34.32**	**73.22**	**59.27**
	GPTQ	46.93	74.24	75.81	36.92	74.59	61.29
	GPTQ + RQ	**47.35**	**75.00**	**76.67**	**37.67**	**75.46**	**61.82**
	GPTAQ	47.27	**75.88**	**78.62**	37.59	75.57	61.97
	GPTAQ + RQ	**50.09**	74.16	77.58	**39.38**	**75.69**	**62.38**

Precision	Method	LAMBADA	OBQA	PIQA	SIQA	Winogrande	Avg.
FP	–	75.70	45.00	80.90	46.88	73.24	66.19
W4A16	RTN	73.78	43.00	77.91	45.04	70.40	61.78
	RTN + RQ	**73.86**	**43.20**	**79.16**	**45.39**	**72.93**	**64.04**
	AWQ	**74.36**	**44.80**	**80.09**	45.75	71.82	63.99
	AWQ + RQ	73.92	44.20	79.43	**46.42**	**72.30**	**64.81**
	GPTQ	74.99	**44.60**	79.49	45.80	**73.56**	65.22
	GPTQ + RQ	**75.12**	**44.60**	**79.98**	**46.06**	72.22	**65.55**
	GPTAQ	75.28	44.40	80.20	45.85	**73.48**	65.50
	GPTAQ + RQ	**75.49**	**45.20**	**80.36**	**46.21**	72.30	**65.76**
W4A4	RTN	51.31	37.80	70.51	42.37	64.25	53.33
	RTN + RQ	**70.02**	**44.00**	**77.58**	**45.65**	**70.72**	**61.94**
	AWQ	60.59	37.80	71.11	41.50	64.01	53.72
	AWQ + RQ	**66.93**	**42.40**	**75.90**	**44.73**	**67.32**	**59.27**
	GPTQ	70.13	**42.60**	76.77	45.29	**69.61**	61.29
	GPTQ + RQ	**71.10**	42.00	**78.07**	**45.60**	68.43	**61.82**
	GPTAQ	70.50	42.00	78.07	45.70	68.51	61.97
	GPTAQ + RQ	**70.99**	42.00	**78.51**	**45.75**	**69.61**	**62.38**

Table 16: Zero-shot accuracy (%) on Qwen3-14B with QuaRot at W4A16 and W4A4. The two panels report complementary sets of tasks for the same runs. Avg. is the ten-task mean.

Precision	Method	ARC-C	ARC-E	BoolQ	CEval	HellaSwag	Avg.
FP	–	60.32	82.95	89.30	82.32	78.82	71.24
W4A16	RTN	55.20	78.32	86.45	78.38	75.82	68.03
	RTN + RQ	**59.90**	**82.45**	**88.47**	**79.20**	**77.72**	**70.54**
	AWQ	57.76	81.86	88.07	78.16	76.17	69.17
	AWQ + RQ	**59.30**	**82.07**	**89.60**	**80.46**	**77.74**	**70.46**
	GPTQ	**59.73**	**82.03**	88.04	79.94	**77.97**	70.52
	GPTQ + RQ	59.47	81.40	**88.96**	**81.13**	77.92	**70.69**
	GPTAQ	58.96	**82.32**	**89.02**	**80.31**	77.45	70.49
	GPTAQ + RQ	**60.07**	82.11	88.47	80.01	**77.76**	**70.53**
W4A4	RTN	48.72	73.40	84.98	74.15	72.26	64.97
	RTN + RQ	**54.95**	**79.92**	**86.79**	**74.67**	**74.45**	**67.61**
	AWQ	51.37	74.20	85.63	73.55	73.26	65.35
	AWQ + RQ	**57.85**	**80.26**	**87.06**	**76.30**	**74.10**	**67.64**
	GPTQ	56.83	77.48	**86.70**	76.37	**74.69**	67.95
	GPTQ + RQ	**57.25**	**79.92**	86.39	**77.04**	74.57	**68.03**
	GPTAQ	55.20	78.41	86.73	74.00	**74.25**	67.02
	GPTAQ + RQ	**55.89**	**79.63**	**87.77**	**76.45**	74.08	**67.53**

Precision	Method	LAMBADA	OBQA	PIQA	SIQA	Winogrande	Avg.
FP	–	67.82	46.40	79.87	52.05	72.53	71.24
W4A16	RTN	67.09	42.20	79.11	48.00	69.69	68.03
	RTN + RQ	**67.75**	**46.60**	**80.03**	**50.61**	**72.69**	**70.54**
	AWQ	62.72	**45.80**	79.33	50.56	71.27	69.17
	AWQ + RQ	**66.25**	45.40	**79.71**	**51.94**	**72.14**	**70.46**
	GPTQ	68.29	45.00	79.43	51.69	**73.09**	70.52
	GPTQ + RQ	**68.35**	**46.00**	**79.82**	**51.79**	72.06	**70.69**
	GPTAQ	67.79	**45.40**	**80.36**	51.28	**71.98**	70.49
	GPTAQ + RQ	**68.50**	44.80	79.87	**51.74**	**71.98**	**70.53**
W4A4	RTN	64.20	41.40	76.55	46.21	67.80	64.97
	RTN + RQ	**65.46**	**43.20**	**77.58**	**49.85**	**69.22**	**67.61**
	AWQ	62.22	42.40	77.09	47.65	66.14	65.35
	AWQ + RQ	**63.50**	**43.60**	**77.48**	**48.72**	**67.56**	**67.64**
	GPTQ	65.38	**44.80**	78.18	**48.77**	**70.32**	67.95
	GPTQ + RQ	**65.79**	43.80	**78.67**	48.21	68.67	**68.03**
	GPTAQ	64.23	**42.40**	77.15	**48.82**	**68.98**	67.02
	GPTAQ + RQ	**65.92**	41.00	**78.29**	48.16	68.11	**67.53**

Table 17: Zero-shot accuracy (%) on Llama-3 70B at W4A4 with QuaRot (Ashkboos et al., 2024) rotation. The two panels cover complementary task sets from the same runs. **Avg.** is the mean over ten tasks; the better entry of each pair is bolded. Table 8 in Appendix B.5 reports the corresponding KL/PPL.

Method	ARC-C	ARC-E	BoolQ	CEval	HellaSwag	Avg.
FP	64.25	85.94	85.38	64.64	84.96	72.92
RTN	22.18	33.63	43.76	23.11	31.10	32.78
RTN + RQ	**55.12**	**80.13**	**84.50**	**46.43**	**79.83**	**66.93**
AWQ	34.04	57.41	64.37	23.11	57.46	47.53
AWQ + RQ	**54.35**	**80.81**	**83.36**	**44.87**	**78.13**	**66.39**
GPTQ	43.77	69.02	72.63	30.91	71.68	58.09
GPTQ + RQ	**54.18**	**79.29**	**82.84**	**47.10**	**77.74**	**66.07**
GPTAQ	**55.72**	79.84	81.96	46.51	78.79	66.20
GPTAQ + RQ	55.03	**80.43**	**84.40**	**47.77**	**79.15**	**67.01**

Method	LAMBADA	OBQA	PIQA	SIQA	Winogrande	Avg.
FP	79.33	49.20	84.39	50.61	80.51	72.92
RTN	7.96	25.40	55.44	35.47	49.72	32.78
RTN + RQ	**77.97**	**44.40**	**80.96**	**46.16**	**73.80**	**66.93**
AWQ	36.31	35.60	68.82	39.25	58.96	47.53
AWQ + RQ	**75.94**	**45.20**	**80.69**	**45.65**	**74.90**	**66.39**
GPTQ	63.05	42.00	75.84	41.40	70.56	58.09
GPTQ + RQ	**76.25**	**44.00**	**79.71**	**46.42**	**73.16**	**66.07**
GPTAQ	73.74	43.80	**80.85**	45.45	**75.37**	66.20
GPTAQ + RQ	**76.77**	**44.20**	**80.85**	**46.16**	75.30	**67.01**

Table 18: Zero-shot accuracy (%) for RTN, AWQ, GPTQ, and GPTAQ with and without ReQuant on Llama-3 8B at W3A4 and W2A4 with QuaRot (Ashkboos et al., 2024) rotation. The two panels cover complementary task sets from the same runs. **Avg.** is the arithmetic mean of the ten reported task scores, with C-Eval counted as one aggregate; the better entry of each pair is bolded. Corresponding KL/PPL are in Table 6 (Appendix B.3).

Precision	Method	ARC-C	ARC-E	BoolQ	CEval	HellaSwag	Avg.
W3A4	RTN	24.83	38.89	52.14	24.07	37.87	36.41
	RTN + RQ	**42.15**	**69.32**	**76.94**	**31.05**	**69.82**	**58.26**
	AWQ	22.95	36.53	46.73	24.29	35.60	35.53
	AWQ + RQ	**41.13**	**68.10**	**73.79**	**29.35**	**69.08**	**56.25**
	GPTQ	40.02	65.82	74.53	**31.43**	**70.08**	57.10
	GPTQ + RQ	**44.11**	**68.73**	**77.16**	31.20	69.79	**58.61**
	GPTAQ	42.92	**69.74**	**74.46**	32.32	**71.26**	58.65
	GPTAQ + RQ	**44.03**	69.70	74.04	**33.66**	70.73	**58.71**
W2A4	RTN	**25.68**	25.17	37.83	**24.52**	26.64	30.18
	RTN + RQ	24.66	**34.26**	**60.43**	23.25	**38.49**	**37.86**
	AWQ	**27.05**	26.18	45.47	24.67	26.36	31.44
	AWQ + RQ	21.59	**32.66**	**57.86**	**26.45**	**32.57**	**34.08**
	GPTQ	23.63	34.18	56.64	**24.22**	35.67	35.88
	GPTQ + RQ	**28.16**	**40.82**	**60.43**	22.66	**45.15**	**41.00**
	GPTAQ	25.94	39.56	60.95	**25.04**	46.17	40.90
	GPTAQ + RQ	**26.88**	**42.13**	**62.08**	23.18	**46.40**	**41.91**

Precision	Method	LAMBADA	OBQA	PIQA	SIQA	Winogrande	Avg.
W3A4	RTN	6.13	31.60	59.68	35.62	53.28	36.41
	RTN + RQ	**66.00**	**40.80**	**75.19**	**42.84**	**68.51**	**58.26**
	AWQ	15.95	25.20	58.76	37.00	52.33	35.53
	AWQ + RQ	**59.23**	**40.40**	**73.94**	**43.04**	**64.40**	**56.25**
	GPTQ	64.53	38.60	**74.92**	**44.11**	66.93	57.10
	GPTQ + RQ	**67.40**	**41.20**	74.76	43.65	**68.11**	**58.61**
	GPTAQ	66.80	42.40	**75.19**	**44.42**	67.01	58.65
	GPTAQ + RQ	**66.83**	**42.80**	74.86	42.53	**67.96**	**58.71**
W2A4	RTN	0.00	28.40	50.71	33.32	49.49	30.18
	RTN + RQ	**18.51**	**30.80**	**57.62**	**36.13**	**54.46**	**37.86**
	AWQ	0.00	**29.60**	50.92	32.70	**51.46**	31.44
	AWQ + RQ	**6.35**	24.40	**54.35**	**34.65**	49.88	**34.08**
	GPTQ	12.32	28.20	55.98	34.80	53.12	35.88
	GPTQ + RQ	**29.23**	**29.00**	**60.77**	**37.46**	**56.35**	**41.00**
	GPTAQ	27.46	**31.00**	61.32	36.49	55.09	40.90
	GPTAQ + RQ	**30.18**	30.20	**61.59**	**38.54**	**57.93**	**41.91**

Table 19: Cost–benefit of the number of refinement sweeps T under W4A16 on Llama-3 8B (K=2). T=0 is the initializer without ReQuant. Quality entries at T=0, 4 reuse Tables 1 and 4; other sweep measurements and all timings are retained. W2/UC/NM denote WikiText-2, UltraChat-2k, and NuminaMath. Time is end-to-end offline quantization wall-clock. Within each setting, the best entry in each column is bolded.

		W2		UC		NM			
Setting	T	KL↓	PPL↓	KL↓	PPL↓	KL↓	PPL↓	Acc.↑	Time (min)↓
RTN+QuaRot	0	0.174	7.63	0.116	3.94	0.100	3.05	61.78	**5.03**
	1	0.067	6.61	0.050	3.65	0.054	2.87	64.83	50.82
	2	0.061	6.56	0.048	3.65	**0.047**	2.86	65.13	60.00
	4	0.079	6.71	0.063	3.70	0.068	2.91	64.04	81.80
	8	**0.052**	**6.51**	**0.040**	**3.62**	0.048	**2.85**	**65.22**	125.24
GPTQ+QuaRot	0	0.054	6.53	0.038	3.59	0.041	**2.83**	65.22	**31.62**
	1	0.044	6.44	0.034	3.59	0.038	**2.83**	65.41	60.58
	2	0.043	6.44	0.033	3.59	0.040	**2.83**	**65.66**	70.86
	4	0.049	6.49	0.036	3.59	0.041	2.84	65.55	94.05
	8	**0.040**	**6.42**	**0.032**	**3.58**	**0.037**	**2.83**	65.49	209.57

C.1 Refinement sweep details

We study how the refinement sweep count T affects ReQuant. Figure 2 reports WikiText-2 perplexity, WikiText-2 KL divergence to the full-precision model, and average zero-shot accuracy on Llama-3 8B for three representative settings: RTN at W4A16 without QuaRot, RTN+QuaRot at W4A16, and GPTAQ+QuaRot at W4A4. All other settings follow Section 4.1. We vary $T \in \{0, 1, \ldots, 8\}$, where $T = 0$ denotes the initial PTQ output. The PPL and KL plots also include full-precision and W4A16 GPTAQ results as horizontal references.

Figure 2 shows large early reductions in RTN PPL/KL, followed by smaller changes with additional sweeps. Mean accuracy fluctuates with T, so the monotone calibration-loss decrease in Section 3.3 should not be read as a guarantee of monotone downstream improvement. The two RTN curves associate QuaRot with lower PPL/KL and generally higher mean accuracy over the tested range, without establishing a general convergence-rate advantage. The GPTAQ curve shows smaller further gains at W4A4. Because that curve uses a different activation precision, its absolute values should not be used to rank GPTAQ against the W4A16 RTN pipelines.

Provenance of the cost summary. Table 19 combines main-table quality entries at T=0, 4 with the remaining sweep measurements and separately recorded timings, as specified in its caption. It is not a single matched sweep record; its rows should not be used to attribute every quality difference solely to T or to infer a per-run quality–time frontier.