KLIQ: KL-INFORMED LOW-BIT QUANTIZATION VIA ATTENTION-DISTRIBUTION-AWARE FISHER-KRONECKER CURVATURE

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training quantization (PTQ) reduces the memory and computation costs of large language models (LLMs), but preserving accuracy at very low bit-widths remains challenging. Existing reconstruction-based curvature approximations in second-order PTQ can overlook the distribution-dependent sensitivity of attention probabilities to query and key quantization errors. We introduce KLIQ, a backpropagation-free PTQ method that uses KL divergence between full-precision and quantized attention distributions to guide curvature estimation, quantization parameter selection, and error compensation. KLIQ introduces three key innovations: (1) a Fisher-Kronecker curvature surrogate that maps a diagonal approximation of the attention-score categorical Fisher to query and key weight spaces; (2) staged functional calibration that conditions key quantization parameter selection on already quantized and compensated queries to account for query-key error interactions; and (3) KL-derived compensation statistics for compensating quantization errors propagated from preceding layers. Extensive experiments across multiple model families show that, when combined with conventional outlier-suppression techniques, KLIQ achieves state-of-the-art performance on language modeling and downstream tasks under low-bit weight-only and weight-activation quantization. The source code is available at `https://anonymous.4open.science/r/KLIQ-7440/`.

1 INTRODUCTION

Deploying large language models (LLMs) remains constrained by their memory footprint and inference cost. Post-training quantization (PTQ) addresses these challenges by representing pretrained weights and activations at reduced precision using a small calibration set (Hubara et al., 2020; Frantar et al., 2023). Recent advances have made low-bit inference increasingly practical, but preserving model quality under aggressive compression, particularly with 2-bit weights, remains challenging (Ashkboos et al., 2024; Chee et al., 2023). At such low precision, quantization errors in the query and key projections can distort attention distributions, changing how the model aggregates information across tokens (Zeng et al., 2025; Kim et al., 2025).

Existing PTQ methods improve low-bit accuracy through complementary strategies. Transformation methods reshape weight and activation distributions to reduce quantization difficulty, using channel scaling Xiao et al. (2023), rotation (Ashkboos et al., 2024), or learned orthogonal and scaling transformations (Hu et al., 2025). Alongside these transformations, second-order methods use curvature information to guide weight quantization and compensate for rounding errors. GPTQ derives this information from layer-wise output reconstruction (Frantar et al., 2023). BoA extends reconstruction to account for dependencies within attention modules (Kim et al., 2025), while TurboBoA accelerates this framework through joint output-channel quantization and further improves error compensation and quantization-parameter optimization (Kim et al., 2026b). However, the relaxed query and key curvature approximations used by BoA and TurboBoA do not retain the distribution-dependent sensitivity of softmax. Attention-score perturbations of similar magnitude can produce different changes in attention probabilities depending on the underlying attention distribution, a distinction these approximations do not explicitly capture.

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.

We introduce KLIQ, a KL-informed PTQ method that preserves attention distributions to guide query and key quantization without backpropagation. KLIQ uses the KL divergence from full-precision attention distributions to their quantized counterparts as its guiding objective, linking curvature estimation, staged calibration, and propagated-error compensation. Our main innovations are:

- **Attention-distribution-aware Fisher-Kronecker curvature.** We derive a weight-space curvature surrogate from the categorical Fisher in attention-score space. Through a diagonal approximation, we obtain Kronecker factors that capture distribution-dependent sensitivity and guide query and key quantization. The formulation supports rotary position embeddings (RoPE) (Su et al., 2024) and optional post-projection query/key root mean square normalization (RMSNorm) (Yang et al., 2025).
- **Staged functional calibration.** We select quantization scales and zero points by minimizing attention KL divergence on a calibration subset, then hold them fixed during weight quantization and compensation. We process the query projection using full-precision keys, followed by the key projection using the quantized and compensated queries, capturing query-key error interactions without exhaustive joint search.
- **KL-derived propagated-error compensation.** We adapt propagated-error compensation to the KL-informed curvature surrogate, incorporating attention sensitivity into the correction statistics to align compensation with the curvature used for quantization.

We evaluate KLIQ on LLaMA, Qwen, and Mistral models under 2-bit weight-only quantization, and on LLaMA models under weight-activation quantization. Combined with QuaRot or OSTQuant, KLIQ consistently improves perplexity, average zero-shot accuracy, and five-shot MMLU over matched TurboBoA baselines across all evaluated models and quantization settings, reducing perplexity by up to 25.84% and improving average zero-shot accuracy and MMLU by up to 2.23 and 5.35 percentage points, respectively.

2 Related Works

Low-Bit LLM Post-Training Quantization Post-training quantization (PTQ) methods for LLMs use activation-aware scaling, reconstruction-based optimization, and representation transformations to preserve accuracy at reduced precision. SmoothQuant migrates quantization difficulty from activations to weights through channel scaling (Xiao et al., 2023), while AWQ uses activation statistics to identify and protect salient weight channels through equivalent scaling (Lin et al., 2024). OmniQuant jointly optimizes weight clipping thresholds and equivalent transformations through block-wise reconstruction (Shao et al., 2024). Rotation-based approaches further facilitate quantization: QuaRot employs randomized Hadamard transformations to suppress outliers (Ashkboos et al., 2024), SpinQuant learns rotations to improve quantized model accuracy (Liu et al., 2025), and OSTQuant combines learnable orthogonal and scaling transformations to improve weight and activation distributions for quantization (Hu et al., 2025). For very-low-bit weight quantization, QuIP combines incoherence processing with adaptive rounding that minimizes a quadratic proxy objective (Chee et al., 2023), while QuIP# combines Hadamard-based incoherence processing with lattice-based vector quantization (Tseng et al., 2024). KLIQ complements these transformation-based approaches, and we evaluate it within the QuaRot and OSTQuant pipelines.

Second-Order and Attention-Aware Quantization Second-order quantization methods use curvature information to guide weight quantization and compensate for quantization errors. GPTQ uses the Hessian of a layer-wise output-reconstruction objective to update unquantized weights (Frantar et al., 2023). GPTAQ extends this framework through asymmetric calibration, accounting for input deviations caused by preceding quantized layers (Li et al., 2025). BoA derives attention-aware Hessians from attention-output reconstruction to capture dependencies among projections within attention modules. To reduce computational complexity, it introduces a relaxed surrogate that avoids explicitly computing the softmax Jacobian (Kim et al., 2025). TurboBoA adopts these Hessian approximations and accelerates quantization by jointly processing multiple output channels, while incorporating propagated-error compensation and quantization-parameter refinement (Kim et al., 2026b). Within this sequential quantization framework, KLIQ instead derives the query- and key-

projection curvature from the KL divergence of attention distributions, capturing the distribution-dependent sensitivity of attention probabilities to quantization errors.

Attention-Distribution and Fisher-Aware Quantization Preserving attention distributions has been explored in both quantization-aware training and post-training quantization. Kim et al. (2022) demonstrate that KL-based matching of post-softmax attention maps improves attention recovery compared with mean-squared-error matching of pre-softmax scores in quantized Transformer encoders. For LLM PTQ, ABQ-LLM observes quantization-induced attention-map distortion and incorporates a symmetric attention-KL loss into block-wise calibration (Zeng et al., 2025). Fisher information has also been employed to construct quantization objectives. BRECQ uses a diagonal Fisher approximation to weight block-reconstruction errors (Li et al., 2021), while FIMA-Q derives an approximate Fisher-based reconstruction objective from output-level KL divergence for Vision Transformer quantization (Wu et al., 2025). DASH-Q reports improved ultra-low-bit LLM quantization using a diagonal approximation to reconstruction curvature, motivated by the sensitivity of off-diagonal Hessian estimates to limited calibration data (Kim et al., 2026a). Kronecker factorization provides a computationally tractable approximation of large Fisher matrices using smaller matrix factors (Martens & Grosse, 2015). KLIQ connects these directions by deriving Kronecker-factored query- and key-weight curvature approximations from the categorical Fisher of attention distributions. Unlike DASH-Q's diagonal approximation to weight-space reconstruction curvature, KLIQ diagonalizes the attention-score Fisher, retaining distribution-dependent per-score sensitivity while yielding generally dense weight-space curvature factors. KLIQ further uses attention KL divergence for staged quantization-parameter calibration and derives input-error statistics for propagated-error compensation, without backpropagation.

3 KLIQ

We now present KLIQ and its three key innovations: attention-distribution-aware Fisher-Kronecker curvature, a staged functional calibration strategy, and KL-derived propagated-error statistics for compensation. Algorithm 1 summarizes the overall procedure, and Figure 1 illustrates the workflow.

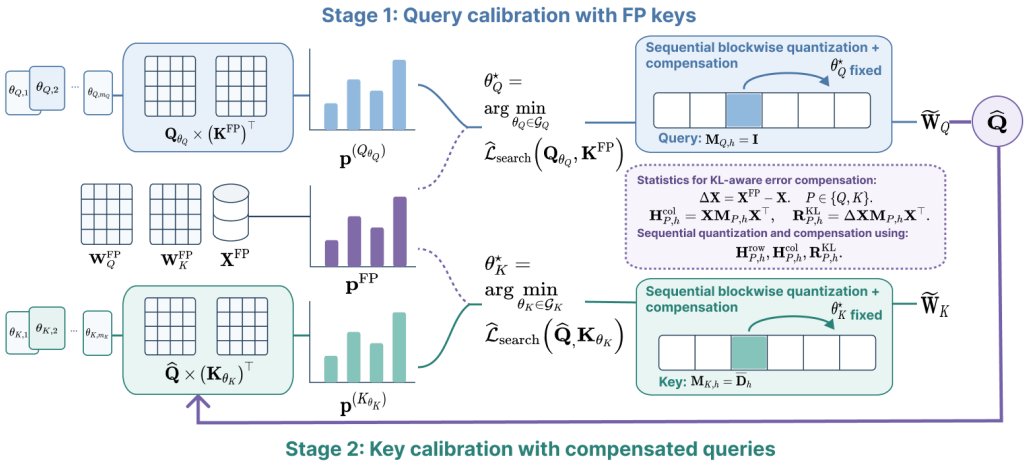

Figure 1: Workflow of KLIQ. Each stage minimizes the search KL loss to select quantization parameters, then fixes them during blockwise quantization and compensation using KL-derived curvature and propagated-error statistics. Query search uses FP keys; key search uses the finalized quantized and compensated queries. Both share the same FP attention reference.

3.1 Attention-Distribution Preservation Objective

KLIQ directly preserves the post-softmax attention distributions induced by quantized query and key projections. For each calibration sequence, we minimize the average KL divergence between

Algorithm 1 KLIQ: KL-informed quantization of query and key projections

Input: Projection weights $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{H\times d_h\times d}$; current and FP input activations $\mathbf{X}, \mathbf{X}^{\mathrm{FP}} \in \mathbb{R}^{d\times T}$; query and key matrices $\mathbf{Q}_h, \mathbf{K}_h \in \mathbb{R}^{T\times d_h}$ for attention head h; calibration set $\mathcal{C}$ and search subset $\mathcal{C}_{\mathrm{search}} \subseteq \mathcal{C}$; quantization parameter grids $\mathcal{G}_Q, \mathcal{G}_K$.

Output: Quantized weights $\widetilde{\mathbf{W}}_Q$ and $\widetilde{\mathbf{W}}_K$.

1: Compute the input discrepancy: $\Delta\mathbf{X} \leftarrow \mathbf{X}^{\mathrm{FP}} - \mathbf{X}$.
2: Compute the diagonal score Fisher $\overline{\mathbf{D}}_h$ using Equation 2, for each head h.
3: Compute the KL-aware Hessian factors:

$$\mathbf{H}^{\mathrm{col}}_{Q,h} = \mathbf{X}\mathbf{X}^\top, \qquad \mathbf{H}^{\mathrm{row}}_{Q,h} = \tfrac{1}{d_h}\mathbf{K}_h^\top\overline{\mathbf{D}}_h\mathbf{K}_h,$$
$$\mathbf{H}^{\mathrm{col}}_{K,h} = \mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top, \quad \mathbf{H}^{\mathrm{row}}_{K,h} = \tfrac{1}{d_h}\mathbf{Q}_h^\top\mathbf{Q}_h.$$

4: Compute the input-deviation correlation matrices:

$$\mathbf{R}^{\mathrm{KL}}_{Q,h} = \Delta\mathbf{X}\mathbf{X}^\top, \qquad \mathbf{R}^{\mathrm{KL}}_{K,h} = \Delta\mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top.$$

SEQQUANTCOMP sequentially quantizes weight blocks and updates the remaining unquantized weights to compensate for quantization and input errors.
5: Select query quantization parameters on $\mathcal{C}_{\mathrm{search}}$ using FP reference keys:

$$\theta_Q^\star \leftarrow \arg\min_{\theta_Q\in\mathcal{G}_Q} \widehat{\mathcal{L}}_{\mathrm{search}}\big(\mathbf{Q}_{\theta_Q}, \mathbf{K}^{\mathrm{FP}}\big).$$

6: Quantize and compensate each query head: ▷ see Algorithm 2

$$\widetilde{\mathbf{W}}_{Q,h} \leftarrow \textsc{SeqQuantComp}\big(\mathbf{W}_{Q,h}, \theta_Q^\star, \mathbf{H}^{\mathrm{col}}_{Q,h}, \mathbf{H}^{\mathrm{row}}_{Q,h}, \mathbf{R}_{Q,h}\big).$$

7: Install $\widetilde{\mathbf{W}}_Q$ and obtain $\widehat{\mathbf{Q}} = \{\mathbf{X}^\top\widetilde{\mathbf{W}}_{Q,h}^\top\}_{h=1}^H$ on $\mathcal{C}_{\mathrm{search}}$.
8: Select key quantization parameters conditioned on the realized queries:

$$\theta_K^\star \leftarrow \arg\min_{\theta_K\in\mathcal{G}_K} \widehat{\mathcal{L}}_{\mathrm{search}}\big(\widehat{\mathbf{Q}}, \mathbf{K}_{\theta_K}\big).$$

9: Quantize and compensate each key head:

$$\widetilde{\mathbf{W}}_{K,h} \leftarrow \textsc{SeqQuantComp}\big(\mathbf{W}_{K,h}, \theta_K^\star, \mathbf{H}^{\mathrm{col}}_{K,h}, \mathbf{H}^{\mathrm{row}}_{K,h}, \mathbf{R}_{K,h}\big).$$

10: Install $\widetilde{\mathbf{W}}_K$.

the full-precision and quantized categorical attention distributions:

$$\mathcal{L}_{\mathrm{KLIQ}} = \frac{1}{HT}\sum_{h=1}^{H}\sum_{t=1}^{T} D_{\mathrm{KL}}\left(\mathbf{p}^{\mathrm{FP}}_{h,t} \,\|\, \mathbf{p}^{\mathrm{quant}}_{h,t}\right), \quad \text{where}$$
$$\mathbf{p}^{c}_{h,t} = \mathrm{softmax}\left(\mathbf{S}^{c}_{h}[t,:]\right), \quad \mathbf{S}^{c}_{h} = \frac{\mathbf{Q}^{c}_{h}(\mathbf{K}^{c}_{h})^\top}{\sqrt{d_h}}, \quad c\in\{\mathrm{FP}, \mathrm{quant}\}. \tag{1}$$

Here, $\mathbf{Q}^c_h, \mathbf{K}^c_h \in \mathbb{R}^{T\times d_h}$ denote the query and key matrices for the attention head h, d_h is the head dimension, and T is the sequence length. The causal attention mask is applied inside the softmax. We use post-softmax KL rather than pre-softmax MSE because it measures distortion directly in the resulting attention probabilities. Its local curvature is the categorical Fisher information, which underlies the curvature construction in Section 3.2.

3.2 DIAGONAL FISHER-KRONECKER CURVATURE

KLIQ constructs curvature surrogates for the query and key projection weights by approximating the categorical Fisher in attention-score space and mapping it to weight space via Kronecker factorization (Martens & Grosse, 2015). It then incorporates architecture-specific transformations, including RoPE and post-projection Q/K RMSNorm.

3.2.1 SCORE-TO-PROJECTION FISHER-KRONECKER CURVATURE

For the attention head h, the score-space KL Hessian is the categorical Fisher information matrix (Martens, 2020). We retain its diagonal as an approximate Fisher, and average it over query positions (see Appendix B.1 for derivation details):

$$\mathbf{g}_h = \frac{1}{T}\sum_{t=1}^{T}\mathbf{p}^{\mathrm{quant}}_{h,t} \odot \left(\mathbf{1} - \mathbf{p}^{\mathrm{quant}}_{h,t}\right), \qquad \overline{\mathbf{D}}_h = \mathrm{Diag}(\mathbf{g}_h), \tag{2}$$

where $\odot$ denotes element-wise multiplication. The diagonal approximation retains distribution-dependent per-score sensitivity while simplifying the construction of the Kronecker factors below and the propagated-error statistics in Section 3.3. The resulting weight-space factors remain generally dense. The ablation in Section 4.3 supports this design choice.

Let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_T] \in \mathbb{R}^{d \times T}$ denote the input activations. Perturbing the query and key projections separately induces the attention-score perturbations:

$$\Delta \mathbf{S}_h^{(Q)} = \frac{1}{\sqrt{d_h}} \mathbf{X}^\top \Delta \mathbf{W}_{Q,h}^\top \mathbf{K}_h^\top, \qquad \Delta \mathbf{S}_h^{(K)} = \frac{1}{\sqrt{d_h}} \mathbf{Q}_h \Delta \mathbf{W}_{K,h} \mathbf{X}. \tag{3}$$

Retaining the diagonal Fisher metric in Equation 2, the second-order KL curvature surrogate is

$$\widetilde{\mathcal{Q}}_h^{(P)} = \frac{1}{2} \operatorname{tr}\Big(\Delta \mathbf{S}_h^{(P)} \overline{\mathbf{D}}_h \big(\Delta \mathbf{S}_h^{(P)}\big)^\top \Big), \qquad P \in \{Q, K\}. \tag{4}$$

Substituting the query-induced score perturbation from Equation 3 into Equation 4 and applying the trace-vectorization identity yields

$$\begin{aligned} \widetilde{\mathcal{Q}}_h^{(Q)} &= \frac{1}{2d_h} \operatorname{tr}\left(\mathbf{X}^\top \Delta \mathbf{W}_{Q,h}^\top \mathbf{K}_h^\top \overline{\mathbf{D}}_h \mathbf{K}_h \Delta \mathbf{W}_{Q,h} \mathbf{X} \right) \\ &= \frac{1}{2} \operatorname{vec}(\Delta \mathbf{W}_{Q,h})^\top \left[\mathbf{X}\mathbf{X}^\top \otimes \frac{1}{d_h} \mathbf{K}_h^\top \overline{\mathbf{D}}_h \mathbf{K}_h \right] \operatorname{vec}(\Delta \mathbf{W}_{Q,h}), \end{aligned} \tag{5}$$

where $\otimes$ denotes the Kronecker product and $\operatorname{vec}(\cdot)$ stacks the columns of a matrix. Likewise, for the key projection:

$$\begin{aligned} \widetilde{\mathcal{Q}}_h^{(K)} &= \frac{1}{2d_h} \operatorname{tr}\left(\mathbf{Q}_h \Delta \mathbf{W}_{K,h} \mathbf{X} \overline{\mathbf{D}}_h \mathbf{X}^\top \Delta \mathbf{W}_{K,h}^\top \mathbf{Q}_h^\top \right) \\ &= \frac{1}{2} \operatorname{vec}(\Delta \mathbf{W}_{K,h})^\top \left[\mathbf{X} \overline{\mathbf{D}}_h \mathbf{X}^\top \otimes \frac{1}{d_h} \mathbf{Q}_h^\top \mathbf{Q}_h \right] \operatorname{vec}(\Delta \mathbf{W}_{K,h}). \end{aligned} \tag{6}$$

Therefore, the resulting approximate Fisher-Kronecker curvature matrices for the query and key projections are

$$\widehat{\mathbf{H}}_{Q,h}^{\mathrm{KL}} = \mathbf{X}\mathbf{X}^\top \otimes \frac{1}{d_h} \mathbf{K}_h^\top \overline{\mathbf{D}}_h \mathbf{K}_h, \qquad \widehat{\mathbf{H}}_{K,h}^{\mathrm{KL}} = \mathbf{X} \overline{\mathbf{D}}_h \mathbf{X}^\top \otimes \frac{1}{d_h} \mathbf{Q}_h^\top \mathbf{Q}_h. \tag{7}$$

The corresponding row and column factors, summarized at line 3 of Algorithm 1, guide sequential weight quantization and error compensation under the KL curvature surrogate. Appendix D details the quantization procedure, while Section 3.3 describes error compensation. Compared with TurboBoA (Kim et al., 2026b), KLIQ incorporates $\overline{\mathbf{D}}_h$ into the query-row and key-column factors, thereby introducing attention-distribution-dependent sensitivity. Appendix B.2 provides the full derivation.

3.2.2 Architecture-Aware Curvature

Let $\widetilde{\mathbf{Q}}_h, \widetilde{\mathbf{K}}_h$ denote the queries and keys after RMSNorm and RoPE. For token t and head h, let $\mathbf{J}_{h,t}^Q, \mathbf{J}_{h,t}^K \in \mathbb{R}^{d_h \times d_h}$ denote the RMSNorm Jacobians evaluated at the corresponding query and key inputs to RMSNorm, and let $\mathbf{R}_t \in \mathbb{R}^{d_h \times d_h}$ denote the RoPE rotation matrix. The KL-curvature approximations are

$$\widehat{\mathbf{H}}_{Q,h}^{\mathrm{KL}} = \mathbf{X}\mathbf{X}^\top \otimes \frac{1}{T d_h} \sum_{t=1}^{T} (\mathbf{J}_{h,t}^Q)^\top \mathbf{R}_t^\top \widetilde{\mathbf{K}}_h^\top \overline{\mathbf{D}}_h \widetilde{\mathbf{K}}_h \mathbf{R}_t \mathbf{J}_{h,t}^Q, \tag{8}$$

$$\widehat{\mathbf{H}}_{K,h}^{\mathrm{KL}} = \mathbf{X} \overline{\mathbf{D}}_h \mathbf{X}^\top \otimes \frac{1}{T d_h} \sum_{t=1}^{T} (\mathbf{J}_{h,t}^K)^\top \mathbf{R}_t^\top \widetilde{\mathbf{Q}}_h^\top \widetilde{\mathbf{Q}}_h \mathbf{R}_t \mathbf{J}_{h,t}^K. \tag{9}$$

Setting $\mathbf{J}_{h,t}^Q = \mathbf{J}_{h,t}^K = \mathbf{I}$ recovers the RoPE-only case. Under the Kronecker approximation, RMSNorm affects only the head-dimensional row factors. Appendix B.3 provides the derivations.

3.3 Staged Functional Calibration and KL-Informed Compensation

Staged Quantization Parameter Search For efficiency, KLIQ searches quantization parameters (scales and zero points) on a fixed calibration subset $\mathcal{C}_{\text{search}}$. Let T_x denote the number of retained tokens in sample x. Suppressing head and sequence indices in the query/key representations, define

$$\widehat{\mathcal{L}}_{\text{search}}(\mathbf{Q},\mathbf{K}) = \frac{1}{N_{\text{search}}} \sum_{x\in\mathcal{C}_{\text{search}}} \sum_{t=1}^{T_x} D_{\text{KL}}\big(\mathbf{p}^{\text{FP}}_{x,t} \parallel \mathbf{p}_{x,t}(\mathbf{Q},\mathbf{K})\big), \quad N_{\text{search}} = \sum_{x\in\mathcal{C}_{\text{search}}} T_x. \tag{10}$$

Here, $\mathbf{p}^{\text{FP}}$ denotes the teacher attention distribution and $\mathbf{p}(\mathbf{Q},\mathbf{K})$ the distribution induced by $\mathbf{Q}$ and $\mathbf{K}$. We select query and key quantization parameters in two stages:

$$\theta_Q^\star = \arg\min_{\theta_Q\in\mathcal{G}_Q} \widehat{\mathcal{L}}_{\text{search}}\big(\mathbf{Q}_{\theta_Q},\mathbf{K}^{\text{FP}}\big), \qquad \theta_K^\star = \arg\min_{\theta_K\in\mathcal{G}_K} \widehat{\mathcal{L}}_{\text{search}}\Big(\widehat{\mathbf{Q}},\mathbf{K}_{\theta_K}\Big), \tag{11}$$

where $\mathcal{G}_Q$ and $\mathcal{G}_K$ are the candidate grids, and $\widehat{\mathbf{Q}}$ denotes the query representations evaluated on $\mathcal{C}_{\text{search}}$ using query weights quantized and compensated with $\theta_Q^\star$. The key search thereby accounts for the query error introduced in the first stage. Ablation study in Section 4.3 supports staged calibration as an effective balance between quantization quality and search efficiency.

KL-Informed Propagated-Error Compensation Functional calibration selects affine quantization parameters but does not correct input error propagated from preceding quantized layers. Let $\mathbf{X},\mathbf{X}^{\text{FP}} \in \mathbb{R}^{d\times T}$ denote the quantized-path and full-precision inputs, respectively, and define $\Delta\mathbf{X} = \mathbf{X}^{\text{FP}} - \mathbf{X}$. Let $\mathbf{M}_{Q,h} = \mathbf{I}$ and $\mathbf{M}_{K,h} = \overline{\mathbf{D}}_h$. For $P \in \{Q,K\}$, the resulting covariance and propagated-error cross-covariance are

$$\mathbf{H}^{\text{col}}_{P,h} = \mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top, \qquad \mathbf{R}^{\text{KL}}_{P,h} = \Delta\mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top. \tag{12}$$

The query-side quantities are shared across heads, whereas the key-side quantities may depend on h. Ignoring the discrete quantization constraint, the stationary correction is

$$\Delta\mathbf{W}^\star_{P,h} = \mathbf{W}_{P,h}\mathbf{R}^{\text{KL}}_{P,h}\big(\mathbf{H}^{\text{col}}_{P,h}\big)^{-1}. \tag{13}$$

Guided by Equation 13, KLIQ uses these curvature and cross-covariance factors in blockwise compensation under discrete quantization, holding the selected quantization parameters fixed. Appendix C derives Equations 12-13.

4 Experiments

4.1 Experimental Setup

Evaluation Protocol Following established practice in low-bit LLM quantization, we apply pre-quantization transformations to suppress weight and activation outliers(Ashkboos et al., 2024; Liu et al., 2025; Hu et al., 2025; Kim et al., 2026b). For weight-only quantization, we use QuaRot and evaluate LLaMA (Touvron et al., 2023), Mistral (Jiang et al., 2023), and Qwen base (Yang et al., 2025) models; for weight-activation quantization, we use OSTQuant and evaluate LLaMA models. All models are calibrated on 128 randomly sampled C4 (Raffel et al., 2020) sequences of 2048 tokens (Frantar et al., 2023; Chee et al., 2023; Kim et al., 2025; Zhang & Shrivastava, 2025). Following standard LLM quantization protocols, we report perplexity (PPL) on the Wiki2 (Merity et al., 2016) and C4 test sets, zero-shot accuracy on eight commonsense reasoning tasks [1], and five-shot accuracy on MMLU (Hendrycks et al., 2020).

Implementation Details All experiments use NVIDIA A100 GPUs (80 GB). For Q/K, KLIQ uses staged functional parameter search (Section 3.3) on eight evenly spaced calibration sequences, retaining the first 1024 tokens of each. Projection inputs are cached, and teacher attention distributions are computed once per search and reused across candidates. The selected parameters remain fixed during compensation, without adaptive search or coordinate-descent (CD) scale refinement. The V/O projections follow TurboBoA, including adaptive search and CD refinement. All four projections are quantized in groups of 16 output channels. We fix the compensation strength at $\alpha = 0.05$ (Li et al., 2025), which gives the best aggregate performance across evaluated models.

[1] PIQA (Bisk et al., 2020), ARC-easy/challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), OpenbookQA (Mihaylov et al., 2018), BoolQ (Clark et al., 2019), and LAMBADA (Paperno et al., 2016).

4.2 Main Results

In this section, we compare KLIQ with established post-training quantization methods, using TurboBoA (Kim et al., 2026b) as the primary baseline under matched hyperparameters. For each method where applicable, we evaluate all four combinations of enabling or disabling row and column activation ordering (Frantar et al., 2023) and report the best results.

Weight-Only Quantization We evaluate 2-bit weight-only quantization on the LLaMA, Qwen, and Mistral families against RTN (Li et al., 2020), GPTQ, and TurboBoA. All methods use QuaRot before quantization. For Qwen3, KLIQ and TurboBoA use the RMSNorm-aware variant from Section 3.2.2.

Across all ten configurations, KLIQ outperforms TurboBoA on every aggregate metric (Table 1), reducing Wiki2 and C4 perplexity by an average of 6.99% and 6.37% and improving average zero-shot accuracy and five-shot MMLU by 0.79 and 2.05 percentage points (pp), respectively. Qwen3-1.7B shows the largest perplexity reductions (23.92% on Wiki2 and 25.84% on C4) and zero-shot gain (+2.23 pp), while Mistral-7B-v0.3 shows the largest MMLU gain (+5.35 pp). KLIQ also wins 59 of 80 individual zero-shot model-task comparisons, including at least five of eight tasks per model (Appendix E.1, Table 5). Among all evaluated quantized methods, KLIQ achieves the lowest perplexity and highest average zero-shot accuracy on every model, and the highest MMLU on nine of ten configurations. The sole exception is MMLU on LLaMA3.2-1B, where RTN scores slightly higher despite near-chance accuracy and catastrophic perplexity. This isolated exception does not undermine KLIQ's state-of-the-art overall performance among the evaluated methods. Additional evaluations against BoA and GPTAQ in Appendix E.2 further demonstrate KLIQ's downstream accuracy gains, while the memory and runtime analysis in Appendix E.3 shows a favorable quality-cost trade-off.

Weight-Activation Quantization We evaluate GPTQ, TurboBoA, and KLIQ on five LLaMA models under W2A4KV16 and W2A4KV4, with all methods using OSTQuant before quantization.

KLIQ outperforms TurboBoA on all four metrics in every configuration (Table 2). Under W2A4KV16, KLIQ reduces Wiki2 and C4 perplexity by an average of 3.98% and 3.59% and improves average zero-shot accuracy and five-shot MMLU by 0.68 and 0.47 pp, respectively. Under W2A4KV4, these gains increase to 6.25% and 4.36% in perplexity and 1.44 and 0.68 pp in accuracy and MMLU. The largest perplexity reductions and zero-shot gain occur on LLaMA3.2-3B under W2A4KV4 (15.41% on Wiki2, 8.81% on C4, and +1.97 pp in accuracy), while LLaMA2-13B achieves the largest MMLU gain (+1.78 pp) under the same setting. KLIQ also wins 57 of 80 individual zero-shot model-task comparisons: 25 of 40 under W2A4KV16 and 32 of 40 under W2A4KV4 (Appendix E.1, Table 6). Among all evaluated quantized methods, KLIQ achieves the lowest perplexity and highest average zero-shot accuracy and MMLU in every configuration.

4.3 Ablation Studies

KLIQ introduces a diagonal Fisher approximation in Section 3.2.1 and staged functional calibration in Section 3.3. To evaluate their respective contributions, we ablate each component while holding all other quantization settings fixed. All ablation experiments are conducted with 2-bit weight-only quantization on LLaMA2 and LLaMA3 models.

Diagonal vs. Full Fisher Table 3 compares the full categorical Fisher with its diagonal approximation in attention-score space. Across all five models, the diagonal variant achieves lower Wiki2 and C4 perplexity, higher average zero-shot and MMLU accuracy, and a 9.4-19.7% reduction in quantization time. The largest improvement occurs on Llama-2-13B, where diagonalization avoids the severe degradation observed with the full Fisher, reducing Wiki2 perplexity from 2.0×10^4 to 10.00 and increasing average accuracy from 38.10% to 59.47%. These gains suggest that diagonalization may provide useful regularization under the large quantization perturbations associated with ultra-low precision, reducing reliance on calibration-dependent curvature couplings. This interpretation is consistent with the motivation of DASH-Q (Kim et al., 2026a), although its diagonal approximation applies to reconstruction curvature rather than the attention-score Fisher.

Table 1: 2-bit weight-only quantization results on LLaMA, Qwen3, and Mistral models

(a) PPL (↓)

Method	LLaMA3.2-1B		LLaMA3.2-3B		LLaMA3-8B		LLaMA2-7B		LLaMA2-13B	
	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4
FP16 Baseline	9.751	12.95	7.814	10.58	6.136	8.889	5.472	7.261	4.884	6.706
RTN	4.2e5	2.9e5	2.8e5	2.1e5	6.9e5	9.6e5	6.9e4	8.1e4	3.0e4	3.4e4
GPTQ	97.14	95.44	45.52	53.54	28.33	32.98	41.09	42.37	16.99	19.10
TurboBoA	55.96	56.97	27.18	30.13	22.10	25.27	13.29	14.93	10.18	11.92
KLIQ	**49.86**	**53.30**	**25.99**	**29.04**	**21.87**	**24.97**	**12.72**	**14.43**	**10.00**	**11.79**
Method	**Qwen3-1.7B**		**Qwen3-4B**		**Qwen3-8B**		**Qwen3-14B**		**Mistral-7B-v0.3**	
	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4
FP16 Baseline	9.388	14.75	7.895	12.76	6.995	11.47	6.379	10.64	5.318	8.156
RTN	2.0e7	2.0e7	2.6e6	1.3e6	2.3e7	4.0e7	7.1e6	7.7e6	6.7e4	8.0e4
GPTQ	56.35	63.53	27.79	33.73	15.05	20.76	14.46	20.00	14.52	18.48
TurboBoA	58.73	62.18	24.24	30.52	16.31	21.95	12.65	17.75	12.16	15.18
KLIQ	**44.68**	**46.11**	**21.47**	**26.81**	**14.97**	**20.36**	**12.42**	**17.49**	**11.90**	**14.98**

(b) Zero-shot accuracy (↑) and five-shot MMLU (↑)

Method	LLaMA3.2-1B		LLaMA3.2-3B		LLaMA3-8B		LLaMA2-7B		LLaMA2-13B	
	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU
FP16 Baseline	57.88	32.10	65.64	56.67	71.04	65.48	67.56	45.93	70.22	55.48
RTN	30.78	**27.09**	31.17	25.17	35.19	26.22	30.31	24.65	30.39	23.32
GPTQ	36.75	25.25	39.13	25.59	44.91	27.48	39.67	26.57	49.64	29.11
TurboBoA	38.74	25.79	47.19	25.85	52.17	26.17	53.40	28.83	59.31	35.11
KLIQ	**38.79**	25.98	**47.78**	**27.77**	**53.00**	**28.50**	**53.97**	**28.89**	**59.47**	**36.38**
Method	**Qwen3-1.7B**		**Qwen3-4B**		**Qwen3-8B**		**Qwen3-14B**		**Mistral-7B-v0.3**	
	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU
FP16 Baseline	62.71	62.70	67.94	73.10	70.62	76.83	73.58	80.62	72.24	62.60
RTN	30.94	23.79	31.31	26.16	31.33	25.30	32.04	24.39	30.89	24.18
GPTQ	40.09	27.17	47.52	32.65	57.04	46.48	58.46	47.88	52.98	30.18
TurboBoA	40.11	24.90	51.44	38.83	58.07	45.16	62.45	56.54	56.71	29.40
KLIQ	**42.34**	**28.56**	**52.55**	**39.02**	**58.08**	**48.28**	**63.70**	**58.91**	**57.76**	**34.75**

Note: All quantization methods use the QuaRot transformation. KLIQ and TurboBoA use the RMSNorm-aware variant for Qwen3 (Base) models.

Table 3: Ablation of full vs. diagonal Fisher

Model	Fisher	PPL ↓		Acc. (%) ↑		Time ↓
		Wiki2	C4	Avg.	MMLU	(min)
Llama-3.2-1B	Full	75.69	77.83	38.62	25.00	16.22
	Diag	**49.86**	**53.30**	**38.79**	**25.98**	**13.02**
Llama-3.2-3B	Full	29.42	33.10	46.32	26.09	42.95
	Diag	**25.99**	**29.04**	**47.78**	**27.77**	**37.24**
Llama-3-8B	Full	23.46	27.64	51.95	27.05	101.4
	Diag	**21.87**	**24.97**	**53.00**	**28.50**	**90.74**
Llama-2-7B	Full	13.62	15.33	53.95	28.43	95.05
	Diag	**12.72**	**14.43**	**53.97**	**28.89**	**84.48**
Llama-2-13B	Full	2.0e4	1.6e4	38.10	29.55	202.7
	Diag	**10.00**	**11.79**	**59.47**	**36.38**	**183.6**

Note: Avg. denotes mean zero-shot accuracy across eight tasks. MMLU is evaluated at five-shot.

Calibration Strategy Table 4 compares four calibration strategies: Staged (KLIQ) calibrates queries against full-precision keys, then keys against compensated, quantized queries; Adaptive repeats parameter search before quantizing each row block; Unstaged selects query and key parameters independently against full-precision counterparts; and None uses default quantizer parameters without search. Staged reduces quantization time by 21-36% relative to Adaptive while achieving higher average zero-shot accuracy on three of five models. Compared with Unstaged, Staged improves 11

Table 2: 2-bit weight-activation quantization results on LLaMA models

(a) PPL (↓)

Precision	Method	LLaMA3.2-1B		LLaMA3.2-3B		LLaMA3-8B		LLaMA2-7B		LLaMA2-13B	
		Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4
FP16	Baseline	9.751	12.95	7.814	10.58	6.136	8.889	5.472	7.261	4.884	6.706
W2A4KV16	GPTQ	125.6	109.9	993.7	766.5	41.64	44.65	52.75	52.63	26.26	27.93
	TurboBoA	58.99	66.37	37.22	41.55	27.44	33.05	13.86	16.19	9.891	11.94
	KLIQ	**56.54**	**65.67**	**35.43**	**38.53**	**25.34**	**31.41**	**13.54**	**15.68**	**9.795**	**11.76**
W2A4KV4	GPTQ	119.3	110.3	1121.3	811.4	56.68	53.71	59.28	53.10	34.44	36.81
	TurboBoA	70.23	77.10	38.81	42.09	27.09	32.10	14.31	16.50	10.47	12.18
	KLIQ	**67.27**	**74.18**	**32.83**	**38.38**	**25.97**	**31.09**	**13.66**	**15.79**	**10.16**	**11.97**

(b) Zero-shot accuracy (↑) and five-shot MMLU (↑)

Precision	Method	LLaMA3.2-1B		LLaMA3.2-3B		LLaMA3-8B		LLaMA2-7B		LLaMA2-13B	
		Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU
FP16	Baseline	57.88	32.10	65.64	56.67	71.04	65.48	67.56	45.93	70.22	55.48
W2A4KV16	GPTQ	33.35	24.41	30.43	24.81	41.78	26.29	37.06	26.39	38.53	26.90
	TurboBoA	38.07	24.98	42.38	26.48	48.61	26.97	49.95	27.08	57.90	33.33
	KLIQ	**38.23**	**25.06**	**43.75**	**27.12**	**49.62**	**27.01**	**50.40**	**27.84**	**58.33**	**34.16**
W2A4KV4	GPTQ	34.27	25.02	30.49	26.04	38.73	26.33	36.23	25.57	39.13	26.65
	TurboBoA	37.06	25.45	41.38	25.73	47.85	26.88	48.33	25.66	55.71	28.76
	KLIQ	**37.63**	**25.97**	**43.35**	**26.34**	**49.34**	**27.16**	**50.17**	**25.85**	**57.04**	**30.54**

Note: All quantization methods use the OSTQuant transformation.

of 20 evaluation metrics at similar runtime, although mean downstream accuracy is broadly comparable: average zero-shot accuracy decreases by 0.03 percentage points, while MMLU accuracy increases by 0.21 points. Relative to None, Staged lowers perplexity in eight of ten model-dataset pairs and improves average zero-shot accuracy on three models, but yields lower MMLU accuracy on all five. Overall, staged calibration offers a practical balance between reconstruction quality and computational cost, without uniformly improving downstream accuracy. Conditioning key calibration on the realized quantized queries accounts for interactions between query and key quantization errors without exhaustive joint search or repeated block-wise calibration.

Table 4: Ablation of staged quantization parameter search strategies

Model	Staged (KLIQ)					Adaptive					Unstaged					None				
	Wiki2	C4	Avg.	M.	Time	Wiki2	C4	Avg.	M.	Time	Wiki2	C4	Avg.	M.	Time	Wiki2	C4	Avg.	M.	Time
LLaMA3.2-1B	49.86	53.30	38.79	25.98	13.02	52.73	55.37	38.95	26.14	18.76	50.29	53.52	39.34	24.46	12.99	49.50	54.24	39.51	26.25	11.12
LLaMA3.2-3B	25.99	29.04	47.78	27.77	37.24	26.23	29.75	47.55	29.25	57.91	25.78	28.95	47.91	27.55	37.15	27.16	28.89	46.59	28.45	33.67
LLaMA3-8B	21.87	24.97	53.00	28.50	90.74	21.47	25.31	51.95	30.22	121.0	21.41	24.78	52.35	29.01	90.70	22.37	25.59	51.86	29.38	85.41
LLaMA2-7B	12.72	14.43	53.97	28.89	84.48	13.03	14.63	53.91	28.72	116.1	12.86	14.48	54.28	29.65	87.36	13.12	14.74	53.31	29.56	77.75
LLaMA2-13B	10.00	11.79	59.47	36.38	183.6	10.07	11.78	60.10	35.32	232.4	10.77	12.50	59.28	35.80	183.3	10.20	11.85	59.51	36.92	173.9

Note: Wiki2/C4: perplexity (↓); Avg./M.: accuracy (%, ↑); Time: minutes (↓). Avg. denotes mean zero-shot accuracy across eight tasks; M. denotes MMLU, evaluated at five-shot.

5 Conclusion

In this work, we present KLIQ, a backpropagation-free PTQ method that uses attention-distribution KL divergence to guide query and key quantization. KLIQ maps a diagonal approximation of the attention-score Fisher to Kronecker-factored weight-space curvature, retaining distribution-dependent sensitivity while keeping curvature computation manageable. It combines this curvature with staged functional calibration that accounts for query-key error interactions and KL-derived statistics for propagated-error compensation. Integrated with QuaRot and OSTQuant, KLIQ consistently improves perplexity, average zero-shot accuracy, and five-shot MMLU accuracy over matched TurboBoA baselines across the evaluated weight-only and weight–activation settings. These results support attention-distribution-aware curvature as an effective complement to outlier suppression for low-bit LLM quantization. Reducing the additional offline calibration cost and extending evaluation to larger models and longer contexts are promising directions for future work.

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.

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

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. Quip: 2-bit quantization of large language models with guarantees. *Advances in neural information processing systems*, 36:4396–4429, 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.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.17323.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Xing Hu, Yuan Cheng, Dawei Yang, Zukang Xu, Zhihang Yuan, Jiangyong Yu, Chen Xu, Zhe Jiang, and Sifan Zhou. Ostquant: Refining large language model quantization with orthogonal and scaling transformations for better distribution fitting. *arXiv preprint arXiv:2501.13987*, 2025.

Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Improving post training neural quantization: Layer-wise calibration and integer programming. *arXiv preprint arXiv:2006.10518*, 2020.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Jaemin Kim, Sungkyun Kim, Junyeol Lee, and Jiwon Seo. Robust ultra low-bit post-training quantization via stable diagonal curvature estimate. In *Proceedings of the Sixth European Workshop on Machine Learning and Systems*, pp. 274–285, 2026a.

Junhan Kim, Ho young Kim, Eulrang Cho, Chungman Lee, Joonyoung Kim, and Yongkweon Jeon. Boa: Attention-aware post-training quantization without backpropagation. In *Proceedings of Forty-second International Conference on Machine Learning (ICML)*, 2025. URL https://arxiv.org/abs/2406.13474.

Junhan Kim, Yeo Jeong Park, Seungwoo Son, Chungman Lee, Ho young Kim, Joonyoung Kim, and Yongkweon Jeon. Turboboa: Faster and exact attention-aware quantization without backpropagation. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://arxiv.org/abs/2602.04929.

Minsoo Kim, Sihwa Lee, Suk-Jin Hong, Du-Seong Chang, and Jungwook Choi. Understanding and improving knowledge distillation for quantization aware training of large transformer encoders. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 6713–6725, 2022.

Yuhang Li, Xin Dong, Sai Qian Zhang, Haoli Bai, Yuanpeng Chen, and Wei Wang. Rtn: Reparameterized ternary network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4780–4787, 2020.

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. *arXiv preprint arXiv:2102.05426*, 2021.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. Gptaq: Efficient finetuning-free quantization for asymmetric calibration. In *Proceedings of the Forty-second International Conference on Machine Learning (ICML)*, 2025. URL https://arxiv.org/abs/2504.02692.

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.

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.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015.

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.

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.

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.

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.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

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, 2024.

Zhuguanyu Wu, Shihe Wang, Jiayi Zhang, Jiaxin Chen, and Yunhong Wang. Fima-q: Post-training quantization for vision transformers by fisher information matrix approximation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14891–14900. IEEE, 2025.

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.

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.

Chao Zeng, Songwei Liu, Yusheng Xie, Hong Liu, Xiaojian Wang, Miao Wei, Shu Yang, Fangmin Chen, and Xing Mei. Abq-llm: Arbitrary-bit quantized inference acceleration for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 22299–22307, 2025.

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.

A AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing and to support retrieval and discovery of related work. We did not use generative AI tools for research ideation or execution, generating synthetic datasets, or interpreting results and drawing conclusions.

B ATTENTION FISHER-KRONECKER CURVATURE

This appendix derives the score-space curvature and shows how it induces the curvature approximations for the query and key projection weights used in Section 3.2.1.

B.1 SCORE-SPACE FISHER AND DIAGONAL APPROXIMATION

The diagonal score-space metric in Equation 2 is derived here. We begin with the exact categorical Fisher, retain its diagonal, and then average the resulting sensitivities over query positions.

Consider attention head h and query position t. Let $\mathbf{s}_{h,t} = \mathbf{S}_h^{\text{quant}}[t,:] \in \mathbb{R}^T$ be the quantized score vector after application of the fixed causal mask, and define

$$\mathbf{q}_{h,t} = \mathbf{p}_{h,t}^{\text{quant}} = \text{softmax}(\mathbf{s}_{h,t}). \tag{14}$$

Let $\mathbf{r}_{h,t}$ denote the parameter of the fixed full-precision target distribution. The per-query KL divergence between the two categorical distributions is

$$\ell_{h,t}(\mathbf{s}_{h,t}) = D_{\text{KL}}(\mathbf{r}_{h,t} \| \mathbf{q}_{h,t}) = -\mathbf{r}_{h,t}^\top \log \mathbf{q}_{h,t} + \text{const}. \tag{15}$$

The softmax Jacobian is

$$\frac{\partial \mathbf{q}_{h,t}}{\partial \mathbf{s}_{h,t}} = \text{Diag}(\mathbf{q}_{h,t}) - \mathbf{q}_{h,t}\mathbf{q}_{h,t}^\top. \tag{16}$$

Using $\mathbf{1}^\top \mathbf{r}_{h,t} = 1$, differentiation of Equation 15 gives

$$\nabla_{\mathbf{s}_{h,t}} \ell_{h,t} = \mathbf{q}_{h,t} - \mathbf{r}_{h,t}, \tag{17}$$

Differentiating Equation 17 once more shows that the score-space KL Hessian equals the softmax Jacobian in Equation 16:

$$\nabla^2_{\mathbf{s}_{h,t}} \ell_{h,t} = \frac{\partial \mathbf{q}_{h,t}}{\partial \mathbf{s}_{h,t}} = \text{Diag}(\mathbf{q}_{h,t}) - \mathbf{q}_{h,t}\mathbf{q}_{h,t}^\top. \tag{18}$$

This curvature also admits a Fisher interpretation. For a one-hot vector $\mathbf{y} \sim \text{Cat}(\mathbf{q}_{h,t})$, the score is $\nabla_{\mathbf{s}_{h,t}} \log p(\mathbf{y}; \mathbf{s}_{h,t}) = \mathbf{y} - \mathbf{q}_{h,t}$, so

$$\begin{aligned} \mathbf{F}_{h,t} &= \mathbb{E}_{\mathbf{y} \sim \text{Cat}(\mathbf{q}_{h,t})} \left[(\mathbf{y} - \mathbf{q}_{h,t})(\mathbf{y} - \mathbf{q}_{h,t})^\top \right] \\ &= \text{Diag}(\mathbf{q}_{h,t}) - \mathbf{q}_{h,t}\mathbf{q}_{h,t}^\top = \nabla^2_{\mathbf{s}_{h,t}} \ell_{h,t}. \end{aligned} \tag{19}$$

Thus, the curvature is determined by the quantized attention distribution $\mathbf{q}_{h,t}$, with no explicit dependence on the fixed target $\mathbf{r}_{h,t}$.

To characterize sensitivity to an attention-score perturbation $\Delta\mathbf{s}_{h,t}$, we apply a second-order Taylor expansion to the per-query KL divergence:

$$\begin{aligned} \ell_{h,t}(\mathbf{s}_{h,t} + \Delta\mathbf{s}_{h,t}) = \ell_{h,t}(\mathbf{s}_{h,t}) &+ \left(\mathbf{q}_{h,t} - \mathbf{r}_{h,t}\right)^\top \Delta\mathbf{s}_{h,t} \\ &+ \frac{1}{2}\Delta\mathbf{s}_{h,t}^\top \mathbf{F}_{h,t} \Delta\mathbf{s}_{h,t} + O\big(\|\Delta\mathbf{s}_{h,t}\|_2^3\big)\,. \end{aligned} \tag{20}$$

When $\mathbf{q}_{h,t} = \mathbf{r}_{h,t}$, the linear term vanishes, and the Fisher quadratic gives the leading local change in KL divergence. More generally, KLIQ retains this positive-semidefinite term as a local sensitivity metric and approximates it by its diagonal:

$$\mathbf{F}_{h,t} \approx \mathbf{D}_{h,t} = \text{Diag}(\mathbf{f}_{h,t}), \qquad \mathbf{f}_{h,t} = \mathbf{q}_{h,t} \odot (\mathbf{1} - \mathbf{q}_{h,t})\,, \tag{21}$$

where $\odot$ is the Hadamard product (element-wise product) operator. Each component of $\mathbf{f}_{h,t}$ measures the local softmax sensitivity at one key position. Probabilities near zero or one have low diagonal curvature, whereas intermediate probabilities have higher curvature.

KLIQ uses a head-wise diagonal metric obtained by averaging over query positions,

$$\mathbf{g}_h = \frac{1}{T}\sum_{t=1}^{T}\mathbf{f}_{h,t}, \qquad \overline{\mathbf{D}}_h = \mathrm{Diag}(\mathbf{g}_h). \tag{22}$$

B.2 Projection-Space Curvature

The aggregate score-space quadratic used in Equation 4 follows from the second-order term in Equation 20. Let $\Delta\mathbf{S}_h \in \mathbb{R}^{T\times T}$ collect the score perturbations, with row t corresponding to query position t. After applying the diagonal approximation $\mathbf{F}_{h,t} \approx \mathbf{D}_{h,t}$, replacing the per-query metrics by their shared head-wise approximation $\overline{\mathbf{D}}_h$, and summing over query positions, we obtain the following expression:

$$\widetilde{\mathcal{Q}}_h^{(P)} = \frac{1}{2}\,\mathrm{tr}\Big(\Delta\mathbf{S}_h^{(P)}\overline{\mathbf{D}}_h\big(\Delta\mathbf{S}_h^{(P)}\big)^\top\Big), \qquad P\in\{Q,K\}. \tag{23}$$

To express Equation 23 in terms of projection-weight perturbations, consider the attention-score map:

$$\mathbf{S}_h = \frac{1}{\sqrt{d_h}}\mathbf{Q}_h\mathbf{K}_h^\top.$$

Let $\mathbf{X}\in\mathbb{R}^{d\times T}$ contain the input activations, let $\mathbf{Q}_h,\mathbf{K}_h\in\mathbb{R}^{T\times d_h}$ be the head-wise query and key matrices, and let $\Delta\mathbf{W}_{Q,h}$ and $\Delta\mathbf{W}_{K,h}$ denote projection-weight perturbations. Holding the opposite projection fixed and using

$$\Delta\mathbf{Q}_h = \mathbf{X}^\top\Delta\mathbf{W}_{Q,h}^\top, \qquad \Delta\mathbf{K}_h^\top = \Delta\mathbf{W}_{K,h}\mathbf{X},$$

gives the induced score perturbations:

$$\Delta\mathbf{S}_h^{(Q)} = \frac{1}{\sqrt{d_h}}\mathbf{X}^\top\Delta\mathbf{W}_{Q,h}^\top\mathbf{K}_h^\top, \qquad \Delta\mathbf{S}_h^{(K)} = \frac{1}{\sqrt{d_h}}\mathbf{Q}_h\Delta\mathbf{W}_{K,h}\mathbf{X}. \tag{24}$$

These are the score perturbations stated in Equation 3.

For the query projection, substituting the first expression in Equation 24 into Equation 23 gives

$$\begin{aligned}\widetilde{\mathcal{Q}}_h^{(Q)} &= \frac{1}{2d_h}\,\mathrm{tr}\left(\mathbf{X}^\top\Delta\mathbf{W}_{Q,h}^\top\mathbf{K}_h^\top\overline{\mathbf{D}}_h\mathbf{K}_h\Delta\mathbf{W}_{Q,h}\mathbf{X}\right)\\ &= \frac{1}{2d_h}\,\mathrm{tr}\left(\Delta\mathbf{W}_{Q,h}^\top\mathbf{K}_h^\top\overline{\mathbf{D}}_h\mathbf{K}_h\Delta\mathbf{W}_{Q,h}\mathbf{X}\mathbf{X}^\top\right).\end{aligned} \tag{25}$$

Here, $\mathrm{vec}(\cdot)$ stacks the columns of a matrix. Using the vectorization identity

$$\mathrm{vec}(\mathbf{AWB}) = (\mathbf{B}^\top\otimes\mathbf{A})\,\mathrm{vec}(\mathbf{W})$$

together with the Frobenius inner-product identity

$$\mathrm{tr}(\mathbf{U}^\top\mathbf{V}) = \mathrm{vec}(\mathbf{U})^\top\,\mathrm{vec}(\mathbf{V}),$$

we obtain

$$\mathrm{tr}(\Delta\mathbf{W}^\top\mathbf{A}\Delta\mathbf{W}\mathbf{B}) = \mathrm{vec}(\Delta\mathbf{W})^\top(\mathbf{B}^\top\otimes\mathbf{A})\,\mathrm{vec}(\Delta\mathbf{W}). \tag{26}$$

Applying Equation 26 to Equation 25 yields

$$\widetilde{\mathcal{Q}}_h^{(Q)} = \frac{1}{2}\,\mathrm{vec}(\Delta\mathbf{W}_{Q,h})^\top\left[\mathbf{X}\mathbf{X}^\top\otimes\frac{1}{d_h}\mathbf{K}_h^\top\overline{\mathbf{D}}_h\mathbf{K}_h\right]\mathrm{vec}(\Delta\mathbf{W}_{Q,h}). \tag{27}$$

Equation 27 has the canonical quadratic form:

$$\widetilde{\mathcal{Q}}_h^{(Q)} = \frac{1}{2}\,\mathrm{vec}(\Delta\mathbf{W}_{Q,h})^\top\widehat{\mathbf{H}}_{Q,h}^{\mathrm{KL}}\,\mathrm{vec}(\Delta\mathbf{W}_{Q,h}) \tag{28}$$

Therefore, the approximate Fisher-Kronecker curvature is

$$\widehat{\mathbf{H}}_{Q,h}^{\mathrm{KL}} = \underbrace{\mathbf{X}\mathbf{X}^\top}_{\mathbf{H}_Q^{\mathrm{col}}} \otimes \underbrace{\frac{1}{d_h}\mathbf{K}_h^\top \overline{\mathbf{D}}_h \mathbf{K}_h}_{\mathbf{H}_{Q,h}^{\mathrm{row}}}, \tag{29}$$

where

$$\mathbf{H}_Q^{\mathrm{col}} = \mathbf{X}\mathbf{X}^\top, \qquad \mathbf{H}_{Q,h}^{\mathrm{row}} = \frac{1}{d_h}\mathbf{K}_h^\top \overline{\mathbf{D}}_h \mathbf{K}_h. \tag{30}$$

The column factor has no head index because all query heads use the same input activations.

Similarly, for the key projection, cyclically reordering the trace and applying Equation 26 yields

$$\begin{aligned} \widetilde{\mathcal{Q}}_h^{(K)} &= \frac{1}{2d_h}\,\mathrm{tr}\left(\mathbf{Q}_h \Delta\mathbf{W}_{K,h} \mathbf{X}\overline{\mathbf{D}}_h \mathbf{X}^\top \Delta\mathbf{W}_{K,h}^\top \mathbf{Q}_h^\top\right) \\ &= \frac{1}{2}\,\mathrm{vec}(\Delta\mathbf{W}_{K,h})^\top \left[\mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top \otimes \frac{1}{d_h}\mathbf{Q}_h^\top\mathbf{Q}_h\right] \mathrm{vec}(\Delta\mathbf{W}_{K,h}). \end{aligned} \tag{31}$$

Hence,

$$\widehat{\mathbf{H}}_{K,h}^{\mathrm{KL}} = \underbrace{\mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top}_{\mathbf{H}_{K,h}^{\mathrm{col}}} \otimes \underbrace{\frac{1}{d_h}\mathbf{Q}_h^\top\mathbf{Q}_h}_{\mathbf{H}_{K,h}^{\mathrm{row}}}. \tag{32}$$

$$\mathbf{H}_{K,h}^{\mathrm{col}} = \mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top, \qquad \mathbf{H}_{K,h}^{\mathrm{row}} = \frac{1}{d_h}\mathbf{Q}_h^\top\mathbf{Q}_h. \tag{33}$$

These row and column factors provide the curvature inputs to the sequential quantization procedure in Appendix D. After incorporating the architecture-specific transformations described in Section 3.2.2, the procedure computes upper-triangular Cholesky factors of their inverses. The inverse column-curvature factor controls compensation across input columns within each output-row block, while the inverse row-curvature factor controls propagation to the remaining output-row blocks. This factorized computation avoids forming or inverting the full weight-space curvature matrix $\widehat{\mathbf{H}}_{P,h}^{\mathrm{KL}}$.

B.3 RMSNorm- and RoPE-Aware Curvature Transformation

Let $\mathbf{z}_{h,t} \in \mathbb{R}^{d_h}$ denote a projected query or key before Root Mean Square Normalization (RMSNorm), and let $\boldsymbol{\gamma} \in \mathbb{R}^{d_h}$ be the corresponding learnable scale parameter. Define

$$r_{h,t} = \sqrt{\frac{1}{d_h}\|\mathbf{z}_{h,t}\|_2^2 + \epsilon}, \qquad \mathrm{RMSNorm}(\mathbf{z}_{h,t}) = \boldsymbol{\gamma} \odot \frac{\mathbf{z}_{h,t}}{r_{h,t}}. \tag{34}$$

Its Jacobian with respect to $\mathbf{z}_{h,t}$ is

$$\mathbf{J}_{h,t} = \mathrm{Diag}(\boldsymbol{\gamma}) \left(\frac{\mathbf{I}}{r_{h,t}} - \frac{\mathbf{z}_{h,t}\mathbf{z}_{h,t}^\top}{d_h r_{h,t}^3}\right). \tag{35}$$

This definition of Jacobian applies separately to queries and keys, yielding $\mathbf{J}_{h,t}^Q$ and $\mathbf{J}_{h,t}^K$. Since RMSNorm is a nonlinear function, it is complicated to calculate the exact perturbation output. We thus apply the first-order Talyor series expansion to a pre-RMSNorm perturbation $\Delta\mathbf{z}_{h,t}$, obtaining $\Delta\,\mathrm{RMSNorm}(\mathbf{z}_{h,t}) \approx \mathbf{J}_{h,t}\Delta\mathbf{z}_{h,t}$. As RoPE subsequently uses the rotary matrix $\mathbf{R}_t$ to maps this perturbation, we thus have

$$\Delta\widetilde{\mathbf{z}}_{h,t} = \mathbf{R}_t\mathbf{J}_{h,t}\Delta\mathbf{z}_{h,t}. \tag{36}$$

Consequently, a row-curvature matrix $\mathbf{H}$ in the transformed coordinates pulls back as

$$\mathbf{H} \longmapsto \mathbf{J}_{h,t}^\top \mathbf{R}_t^\top \mathbf{H} \mathbf{R}_t \mathbf{J}_{h,t}. \tag{37}$$

Applying this identity to the query and key row factors from Section 3.2.1, followed by averaging over positions, gives Equation 8 and Equation 9.

C Derivation of KL-Informed Input-Error Compensation

This appendix derives the KL-weighted propagated-error cross-covariance and stationary correction used in Section 3.3. It applies the projection-space Fisher-Kronecker quadratic derived in Appendix B.2 when the current projection receives different activations along the full-precision and quantized execution paths.

Let $\mathbf{X}^{\mathrm{FP}}, \mathbf{X} \in \mathbb{R}^{d\times T}$ be the current-path and full-precision input activations. Define

$$\Delta\mathbf{X} = \mathbf{X}^{\mathrm{FP}} - \mathbf{X}. \tag{38}$$

For projection $P \in \{Q, K\}$ and head h, define

$$\widetilde{\mathbf{W}}_{P,h} = \mathbf{W}_{P,h} + \Delta\mathbf{W}_{P,h}, \qquad \mathbf{M}_{Q,h} = \mathbf{I}, \qquad \mathbf{M}_{K,h} = \overline{\mathbf{D}}_h. \tag{39}$$

The choices of $\mathbf{M}_{Q,h}$ and $\mathbf{M}_{K,h}$ follow directly from the query and key pullbacks in Equations 27 and 31: the diagonal Fisher contracts into the query row factor, whereas on the key side it contracts over token positions and therefore appears in the column factor.

The projection-output discrepancy is

$$\begin{aligned}\mathbf{E}^{\mathrm{out}}_{P,h} &= \mathbf{W}_{P,h}\mathbf{X}^{\mathrm{FP}} - \widetilde{\mathbf{W}}_{P,h}\mathbf{X} \\ &= \mathbf{W}_{P,h}\Delta\mathbf{X} - \Delta\mathbf{W}_{P,h}\mathbf{X}.\end{aligned} \tag{40}$$

Equation 25 and 31 follow from pulling back the score-space quadratic in Equation 23 through the query and key projections, respectively, while holding the opposite projection fixed. In both cases, the resulting quadratic measures the projection-output perturbation $\Delta\mathbf{W}_{P,h}\mathbf{X}$, with row curvature $\mathbf{H}^{\mathrm{row}}_{P,h}$ and token weighting $\mathbf{M}_{P,h}$. When the full-precision and quantized paths receive different inputs, Equation 40 gives the corresponding output discrepancy $\mathbf{E}^{\mathrm{out}}_{P,h}$, which includes both propagated input error and the weight correction. Applying the same quadratic metric to this discrepancy gives

$$\mathcal{J}_{P,h} = \frac{1}{2}\,\mathrm{tr}\big[\mathbf{H}^{\mathrm{row}}_{P,h}\mathbf{E}^{\mathrm{out}}_{P,h}\mathbf{M}_{P,h}(\mathbf{E}^{\mathrm{out}}_{P,h})^\top\big]. \tag{41}$$

When $\Delta\mathbf{X} = 0$, the discrepancy reduces to $-\Delta\mathbf{W}_{P,h}\mathbf{X}$, and Equation 41 recovers Equation 25 and 31, since the sign cancels in the quadratic.

Substituting Equation 40 and expanding yields

$$\begin{aligned}\mathcal{J}_{P,h} = \frac{1}{2}\,&\mathrm{tr}\big[\mathbf{H}^{\mathrm{row}}_{P,h}\mathbf{W}_{P,h}\Delta\mathbf{X}\mathbf{M}_{P,h}\Delta\mathbf{X}^\top\mathbf{W}^\top_{P,h}\big] \\ &- \mathrm{tr}\big[\Delta\mathbf{W}^\top_{P,h}\mathbf{H}^{\mathrm{row}}_{P,h}\mathbf{W}_{P,h}\Delta\mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top\big] \\ &+ \frac{1}{2}\,\mathrm{tr}\big[\Delta\mathbf{W}^\top_{P,h}\mathbf{H}^{\mathrm{row}}_{P,h}\Delta\mathbf{W}_{P,h}\mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top\big].\end{aligned} \tag{42}$$

The first term is independent of $\Delta\mathbf{W}_{P,h}$. Thus, we can treat it as constant. Defining

$$\mathbf{H}^{\mathrm{col}}_{P,h} = \mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top, \qquad \mathbf{R}^{\mathrm{KL}}_{P,h} = \Delta\mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top. \tag{43}$$

The variable-dependent part becomes the difference between the third and second terms, plus the first constant term:

$$\mathcal{J}_{P,h} = \frac{1}{2}\,\mathrm{tr}\big[\Delta\mathbf{W}^\top_{P,h}\mathbf{H}^{\mathrm{row}}_{P,h}\Delta\mathbf{W}_{P,h}\mathbf{H}^{\mathrm{col}}_{P,h}\big] - \mathrm{tr}\big[\Delta\mathbf{W}^\top_{P,h}\mathbf{H}^{\mathrm{row}}_{P,h}\mathbf{W}_{P,h}\mathbf{R}^{\mathrm{KL}}_{P,h}\big] + \mathrm{const}. \tag{44}$$

Since the curvature factors are symmetric, the derivative of Equation 44 is

$$\frac{\partial\mathcal{J}_{P,h}}{\partial\Delta\mathbf{W}_{P,h}} = \mathbf{H}^{\mathrm{row}}_{P,h}\left(\Delta\mathbf{W}_{P,h}\mathbf{H}^{\mathrm{col}}_{P,h} - \mathbf{W}_{P,h}\mathbf{R}^{\mathrm{KL}}_{P,h}\right). \tag{45}$$

On the supported subspace of the row curvature, the stationary condition is therefore

$$\Delta\mathbf{W}_{P,h}\mathbf{H}^{\mathrm{col}}_{P,h} = \mathbf{W}_{P,h}\mathbf{R}^{\mathrm{KL}}_{P,h}, \tag{46}$$

which gives

$$\boxed{\Delta\mathbf{W}^\star_{P,h} = \mathbf{W}_{P,h}\mathbf{R}^{\mathrm{KL}}_{P,h}\big(\mathbf{H}^{\mathrm{col}}_{P,h}\big)^{-1}.} \tag{47}$$

The row factor affects the cost assigned to feasible quantization errors and the subsequent row-block propagation, but is canceled from the unconstrained stationary correction. Substituting $\mathbf{M}_{Q,h} = \mathbf{I}$ and $\mathbf{M}_{K,h} = \overline{\mathbf{D}}_h$ gives

$$\begin{aligned} \mathbf{H}_Q^{\mathrm{col}} &= \mathbf{X}\mathbf{X}^\top, & \mathbf{R}_Q^{\mathrm{KL}} &= \Delta\mathbf{X}\mathbf{X}^\top, \\ \mathbf{H}_{K,h}^{\mathrm{col}} &= \mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top, & \mathbf{R}_{K,h}^{\mathrm{KL}} &= \Delta\mathbf{X}\overline{\mathbf{D}}_h\mathbf{X}^\top. \end{aligned} \tag{48}$$

The query correction remains KL-aware through $\mathbf{H}_{Q,h}^{\mathrm{row}}$, which contains $\overline{\mathbf{D}}_h$. For keys, the column curvature and propagated-error cross-covariance use the same Fisher-derived token weighting. Setting $\mathbf{M}_{P,h} = \mathbf{I}$ recovers the standard TurboBoA statistics and correction.

D SEQUENTIAL BLOCKWISE QUANTIZATION ALGORITHM

Algorithm 2 specifies the SEQQUANTCOMP routine used in Algorithm 1. It follows the sequential quantization and compensation procedure of TurboBoA (Kim et al., 2026b), and uses the KL-derived curvature factors from Section 3.2.1 (detailed in Appendix B) and propagated-error statistics from Section 3.3 (detailed in Appendix C). The routine quantizes input columns of the projection matrices sequentially within each output-row block, then propagates the block's discrepancy to the remaining output rows. We describe the fixed-parameter Q/K path used by the staged calibration procedure: the functionally selected scales and zero points remain fixed throughout quantization and compensation (refer to Section 3.3).

Inputs and notation Fix a projection $P \in \{Q, K\}$ and head h. To simplify the algorithm, write $\mathbf{W} = \mathbf{W}_{P,h} \in \mathbb{R}^{d_h \times d}$, $\mathbf{H}^{\mathrm{col}} = \mathbf{H}^{\mathrm{col}}_{P,h} \in \mathbb{R}^{d \times d}$, $\mathbf{H}^{\mathrm{row}} = \mathbf{H}^{\mathrm{row}}_{P,h} \in \mathbb{R}^{d_h \times d_h}$, and $\mathbf{R}^{\mathrm{KL}} = \mathbf{R}^{\mathrm{KL}}_{P,h} \in \mathbb{R}^{d \times d}$. Here, d and d_h are the input and head dimensions; columns of $\mathbf{W}$ index input channels, and rows index output channels. The row factor includes the applicable RoPE and RMSNorm transformations from Section 3.2.2 (detailed in Appendix B.3). We use the same factor symbols for their numerically stabilized versions supplied to the quantizer.

As defined in Equation 43, $\mathbf{R}^{\mathrm{KL}} = \Delta\mathbf{X}\mathbf{M}_{P,h}\mathbf{X}^\top$, where $\Delta\mathbf{X} = \mathbf{X}^{\mathrm{FP}} - \mathbf{X}$ and $\mathbf{M}_{Q,h} = \mathbf{I}$, $\mathbf{M}_{K,h} = \overline{\mathbf{D}}_h$. Thus, the query cross-covariance is shared across heads, whereas the key cross-covariance uses the same head-specific token weighting as its column curvature. These cross-covariances need not be symmetric.

Let $\theta^\star = (\mathbf{s}, \mathbf{z})$ denote the entries of the selected projection parameters $\theta^\star_P$ corresponding to this head, where $\mathbf{s}, \mathbf{z} \in \mathbb{R}^{d_h}$ contain one positive scale s_i and zero point z_i per output row i. For bit-width b, the quantizer returns weights reconstructed from affine integer codes:

$$\mathrm{Quant}_b(\mathbf{w}; \mathbf{s}, \mathbf{z}) = \mathbf{s} \odot \left[\mathrm{clip}\big(\mathrm{round}(\mathbf{w} \oslash \mathbf{s}) + \mathbf{z}, 0, 2^b - 1\big) - \mathbf{z}\right]. \tag{49}$$

All operations in Equation 49 are elementwise: $\oslash$ and $\odot$ denote division and multiplication, respectively; round rounds to the nearest integer, with ties to even as in the implementation; and clip restricts the codes to $[0, 2^b - 1]$. The returned matrix $\widetilde{\mathbf{W}}$ contains these reconstructed values, rather than the integer codes themselves.

Curvature factors and input-error propagation To use the curvature factors derived in Section 3.2.1 for sequential compensation, we compute upper-triangular factors of their stabilized inverses:

$$\mathbf{U}_c^\top \mathbf{U}_c = (\mathbf{H}^{\mathrm{col}})^{-1}, \qquad \mathbf{U}_r^\top \mathbf{U}_r = (\mathbf{H}^{\mathrm{row}})^{-1}, \tag{50}$$

where $\mathbf{U}_c \in \mathbb{R}^{d \times d}$ and $\mathbf{U}_r \in \mathbb{R}^{d_h \times d_h}$ are upper triangular with positive diagonals. Equivalently, each is the transpose of the lower-triangular Cholesky factor of the corresponding inverse curvature. They govern propagation along input columns and output rows, respectively.

For compensation strength $\alpha \geq 0$, precompute

$$\mathbf{R} = \alpha\mathbf{R}^{\mathrm{KL}}, \qquad \mathbf{P} = \mathrm{triu}_+(\mathbf{R}\mathbf{U}_c^\top)\mathbf{U}_c, \qquad \mathbf{C} = \mathbf{R}\mathbf{U}_c^\top\mathbf{U}_c = \mathbf{R}(\mathbf{H}^{\mathrm{col}})^{-1}. \tag{51}$$

All three matrices are $d \times d$. The operator triu_+ retains only entries strictly above the diagonal. Consequently, $\mathbf{P}$ is strictly upper triangular and directs input-error compensation only to later input columns. The full matrix $\mathbf{C}$ is used when propagating between output-row blocks. Its connection to Appendix C is the correction operator $\alpha\mathbf{R}^{\mathrm{KL}}(\mathbf{H}^{\mathrm{col}})^{-1}$ in Equation 47, evaluated here using the stabilized column factor. The sequential routine uses this operator within its block updates; it does not first add the unconstrained correction $\mathbf{WC}$ to all weights. Setting $\alpha = 0$ removes the input-error terms while retaining curvature-based quantization-error compensation.

Sequential quantization within a row block Initialize the working head weights $\mathbf{A} \leftarrow \mathbf{W}$ and the output $\widetilde{\mathbf{W}} \leftarrow \mathbf{0}$. For a block starting at row i, let $r = \min(i + N - 1, d_h)$, $J = \{i, \ldots, r\}$, and $F = \{r + 1, \ldots, d_h\}$, where the positive integer N is the maximum number of output rows processed jointly. Thus, J indexes the current block and F the unprocessed rows; $|J|$ and $|F|$ denote their sizes, and a colon selects all entries along an axis. Save a copy $\mathbf{B} \leftarrow \mathbf{A}_{J,:} \in \mathbb{R}^{|J| \times d}$ and

initialize a separate working copy $\mathbf{V} \leftarrow \mathbf{B}$. The snapshot $\mathbf{B}$ includes compensation received from earlier row blocks and remains fixed during the inner loop.

For each input column $j = 1, \ldots, d$, compute

$$\mathbf{w} \leftarrow \mathbf{V}_{:,j}, \qquad \mathbf{q} \leftarrow \mathrm{Quant}_b(\mathbf{w}; \mathbf{s}_J, \mathbf{z}_J), \qquad \mathbf{e} \leftarrow \frac{\mathbf{w} - \mathbf{q}}{(\mathbf{U}_c)_{jj}}. \tag{52}$$

Here, $\mathbf{w}, \mathbf{q}, \mathbf{e} \in \mathbb{R}^{|J|}$ are column vectors, and $\mathbf{s}_J, \mathbf{z}_J$ select the fixed parameters for the current rows. The vector $\mathbf{e}$ is the quantization residual normalized by the corresponding inverse-curvature Cholesky diagonal. Store $\widetilde{\mathbf{W}}_{J,j} \leftarrow \mathbf{q}$ and update

$$\mathbf{V}_{:,j:d} \leftarrow \mathbf{V}_{:,j:d} - \mathbf{e}(\mathbf{U}_c)_{j,j:d} + \mathbf{w}\mathbf{P}_{j,j:d}. \tag{53}$$

The notation $j : d$ denotes columns $j, \ldots, d$, and each product on the right is an outer product. Both $\mathbf{w}$ and $\mathbf{e}$ are evaluated before this update. The first term propagates the quantization residual through the column curvature; the second propagates input-error compensation using the current, already compensated column $\mathbf{w}$. Since $P_{jj} = 0$, the update sets $\mathbf{V}_{:,j} = \mathbf{q}$ and changes only later columns thereafter. All rows in J are quantized jointly, without sequential row updates inside the block.

Propagation to the remaining row blocks After all columns in J have been quantized, the row curvature determines how its discrepancy affects F. When $F \neq \varnothing$, solve

$$(\mathbf{U}_r)_{J,J}\mathbf{Z} = (\mathbf{U}_r)_{J,F}, \qquad \mathbf{G} = \mathbf{Z}^\top. \tag{54}$$

The upper-triangular solve gives $\mathbf{Z} \in \mathbb{R}^{|J|\times|F|}$ and $\mathbf{G} \in \mathbb{R}^{|F|\times|J|}$, which maps a discrepancy in the current rows to a correction of the remaining rows. Apply

$$\mathbf{A}_{F,:} \leftarrow \mathbf{A}_{F,:} - \mathbf{G}\left(\mathbf{B} - \widetilde{\mathbf{W}}_{J,:} + \mathbf{B}\mathbf{C}\right). \tag{55}$$

The matrix $\mathbf{B} - \widetilde{\mathbf{W}}_{J,:}$ is the block's weight discrepancy, and $\mathbf{BC}$ is its input-error contribution. Both are computed from the saved block $\mathbf{B}$, not the modified inner-loop weights $\mathbf{V}$. Equation 55 subtracts their combined propagation from the unprocessed rows, matching the implemented update with $\Delta\mathbf{X} = \mathbf{X}^{\mathrm{FP}} - \mathbf{X}$. Previously stored quantized rows remain fixed. Processing continues until all rows have been written to $\widetilde{\mathbf{W}}$.

Algorithm 2 SEQQUANTCOMP: sequential quantization with compensation

Input: Head weights $\mathbf{W} \in \mathbb{R}^{d_h \times d}$; fixed parameters $\theta^\star = (\mathbf{s}, \mathbf{z})$; stabilized curvature factors $\mathbf{H}^{\mathrm{col}} \in \mathbb{R}^{d\times d}$ and $\mathbf{H}^{\mathrm{row}} \in \mathbb{R}^{d_h \times d_h}$; cross-covariance $\mathbf{R}^{\mathrm{KL}} \in \mathbb{R}^{d\times d}$; bit-width b; row-block size N; compensation strength α.

Output: Quantized weights $\widetilde{\mathbf{W}} \in \mathbb{R}^{d_h \times d}$.

Compute $\mathbf{U}_c, \mathbf{U}_r$ using Equation 50.
Compute the input-error propagation matrices:

$$\mathbf{R} \leftarrow \alpha\mathbf{R}^{\mathrm{KL}}, \qquad \mathbf{P} \leftarrow \mathrm{triu}_+(\mathbf{R}\mathbf{U}_c^\top)\mathbf{U}_c, \qquad \mathbf{C} \leftarrow \mathbf{R}\mathbf{U}_c^\top\mathbf{U}_c.$$

Initialize $\mathbf{A} \leftarrow \mathrm{copy}(\mathbf{W})$; $\widetilde{\mathbf{W}} \leftarrow \mathbf{0}$.
for $i = 1, 1+N, 1+2N, \ldots \leq d_h$ **do**
  $r \leftarrow \min(i+N-1, d_h)$; $J \leftarrow \{i, \ldots, r\}$; $F \leftarrow \{r+1, \ldots, d_h\}$.
  $\mathbf{B} \leftarrow \mathrm{copy}(\mathbf{A}_{J,:})$; $\mathbf{V} \leftarrow \mathrm{copy}(\mathbf{B})$.
  for $j = 1, \ldots, d$ **do**
    $\mathbf{w} \leftarrow \mathrm{copy}(\mathbf{V}_{:,j})$; $\mathbf{q} \leftarrow \mathrm{Quant}_b(\mathbf{w}; \mathbf{s}_J, \mathbf{z}_J)$.
    $\widetilde{\mathbf{W}}_{J,j} \leftarrow \mathbf{q}$; $\mathbf{e} \leftarrow (\mathbf{w} - \mathbf{q})/(\mathbf{U}_c)_{jj}$.
    $\mathbf{V}_{:,j:d} \leftarrow \mathbf{V}_{:,j:d} - \mathbf{e}(\mathbf{U}_c)_{j,j:d} + \mathbf{w}\mathbf{P}_{j,j:d}$.
  end for
  if $F \neq \varnothing$ **then**
    Solve $(\mathbf{U}_r)_{J,J}\mathbf{Z} = (\mathbf{U}_r)_{J,F}$; $\mathbf{G} \leftarrow \mathbf{Z}^\top$.
    $\mathbf{A}_{F,:} \leftarrow \mathbf{A}_{F,:} - \mathbf{G}(\mathbf{B} - \widetilde{\mathbf{W}}_{J,:} + \mathbf{B}\mathbf{C})$.
  end if
end for
return $\widetilde{\mathbf{W}}$.

E Additional Experimental Results

E.1 Per-Task Zero-Shot Accuracy Results

Tables 5 and 6 report per-task zero-shot accuracy for KLIQ and the matched TurboBoA baseline. Scores are expressed as percentages, and Avg. denotes the unweighted mean across the eight tasks. Based on the displayed scores, KLIQ improves average accuracy in every evaluated model-precision configuration, with gains distributed across a broad range of tasks.

For 2-bit weight-only quantization, KLIQ outperforms TurboBoA in 59 of the 80 model-task comparisons and ties in two. It wins on at least five of the eight tasks for every model and achieves higher average accuracy across all ten models, demonstrating consistent benefits across the evaluated model families and scales. The gains are particularly pronounced on Qwen3-1.7B, where KLIQ improves all eight tasks, including ARC-Easy (+6.69 pp) and HellaSwag (+3.34 pp). Even on Qwen3-8B, where the average improvement is small, KLIQ achieves higher accuracy on five of the eight tasks.

Under weight-activation quantization, KLIQ outperforms TurboBoA in 57 of the 80 model-task-precision comparisons and ties in four, while improving average accuracy for every model under both precision settings. Its task-level advantage is especially consistent under W2A4KV4: KLIQ wins 32 of 40 comparisons, compared with 25 of 40 under W2A4KV16, with two ties in each setting. Under W2A4KV4, it improves all eight tasks for both LLaMA3.2-3B and LLaMA2-13B. Furthermore, KLIQ matches or exceeds TurboBoA on WinoGrande and LAMBADA in every evaluated weight-activation configuration. These results demonstrate broad task-level improvements alongside consistent gains in average accuracy, including when both activations and the KV cache are quantized to 4 bits.

Table 5: Per-task zero-shot accuracy results of 2-bit weight-only quantization

Model	Method	Zero-shot Accuracy (%) ↑								
		PIQA	ARC-e	ARC-c	HS	WG	OBQA	BQ	LAMB	**Avg.**
LLaMA3.2-1B	TurboBoA	60.83	37.75	21.67	34.45	52.57	26.60	60.28	15.82	38.74
	KLIQ	61.70	38.85	22.53	33.73	52.80	24.40	61.90	14.42	**38.79**
LLaMA3.2-3B	TurboBoA	67.03	49.75	26.79	45.00	58.01	30.80	64.01	36.11	47.19
	KLIQ	66.05	50.25	26.79	45.62	58.56	30.40	64.59	40.00	**47.78**
LLaMA3-8B	TurboBoA	70.73	57.83	31.14	52.06	61.25	33.40	68.32	42.62	52.17
	KLIQ	70.78	57.03	31.74	52.93	61.48	34.60	69.48	45.93	**53.00**
LLaMA2-7B	TurboBoA	71.55	55.30	28.50	54.21	62.67	36.00	67.22	51.80	53.40
	KLIQ	71.06	56.57	30.38	54.81	61.88	35.40	67.37	54.34	**53.97**
LLaMA2-13B	TurboBoA	73.67	61.95	33.96	60.73	65.11	37.80	72.94	68.31	59.31
	KLIQ	74.37	61.95	34.73	61.72	66.38	37.00	70.92	68.68	**59.47**
Qwen3-1.7B	TurboBoA	60.17	38.13	22.61	34.31	52.17	26.60	62.42	24.43	40.11
	KLIQ	62.51	44.82	24.83	37.65	52.64	27.40	63.15	25.71	**42.34**
Qwen3-4B	TurboBoA	66.70	57.03	33.02	45.82	57.77	32.60	64.10	54.47	51.44
	KLIQ	68.93	60.65	35.49	48.52	59.67	34.20	62.35	50.57	**52.55**
Qwen3-8B	TurboBoA	71.93	69.99	41.13	54.31	62.98	34.40	68.53	61.27	58.07
	KLIQ	72.58	70.50	41.38	55.96	64.17	34.20	65.60	60.28	**58.08**
Qwen3-14B	TurboBoA	74.32	71.46	43.43	61.02	69.77	38.60	77.71	63.26	62.45
	KLIQ	75.30	73.32	45.90	61.91	69.22	38.20	79.94	65.83	**63.70**
Mistral-7B-v0.3	TurboBoA	72.96	62.25	33.36	58.33	65.51	33.00	69.42	58.84	56.71
	KLIQ	73.78	63.09	34.81	59.17	67.32	35.00	68.17	60.78	**57.76**

Note: All quantization methods use the QuaRot transformation. KLIQ and TurboBoA use the RMSNorm-aware variant for Qwen3 (Base) models.

Table 6: Per-task zero-shot accuracy results of 2-bit weight-activation quantization

Model	Precision	Method	Zero-shot Accuracy (%) ↑								
			PIQA	ARC-e	ARC-c	HS	WG	OBQA	BQ	LAMB	**Avg.**
LLaMA3.2-1B	W2A4KV16	TurboBoA	60.83	38.26	22.95	33.49	51.54	26.40	58.93	12.13	38.07
		KLIQ	59.30	37.88	21.84	33.30	51.54	27.80	61.77	12.38	**38.23**
	W2A4KV4	TurboBoA	58.38	36.95	21.76	32.71	50.83	26.60	56.97	12.26	37.06
		KLIQ	57.94	36.41	23.63	32.38	51.30	26.60	60.52	12.26	**37.63**
LLaMA3.2-3B	W2A4KV16	TurboBoA	61.26	44.95	25.68	39.15	53.51	28.40	62.42	23.68	42.38
		KLIQ	62.19	46.04	26.96	41.37	54.93	29.60	61.25	27.65	**43.75**
	W2A4KV4	TurboBoA	60.94	42.68	24.57	38.85	53.35	26.00	61.50	23.15	41.38
		KLIQ	62.79	45.45	25.43	40.79	54.30	28.80	62.66	26.59	**43.35**
LLaMA3-8B	W2A4KV16	TurboBoA	67.79	54.42	29.95	48.12	58.01	31.60	64.31	34.66	48.61
		KLIQ	67.85	54.42	31.06	49.28	58.48	31.40	68.10	36.39	**49.62**
	W2A4KV4	TurboBoA	65.23	50.63	27.73	48.51	56.75	32.40	65.54	36.02	47.85
		KLIQ	66.76	51.81	28.84	49.57	58.17	31.20	66.97	41.43	**49.34**
LLaMA2-7B	W2A4KV16	TurboBoA	68.39	48.74	29.35	52.21	56.43	30.20	66.91	47.33	49.95
		KLIQ	68.28	48.57	29.01	53.26	59.12	31.00	64.98	48.98	**50.40**
	W2A4KV4	TurboBoA	67.79	50.17	28.41	51.53	53.35	30.00	61.50	43.92	48.33
		KLIQ	66.38	49.07	28.92	52.81	57.22	33.40	65.11	48.44	**50.17**
LLaMA2-13B	W2A4KV16	TurboBoA	71.44	61.11	34.04	61.33	63.30	33.80	72.08	66.06	57.90
		KLIQ	71.00	60.77	35.15	62.54	63.69	33.20	73.39	66.87	**58.33**
	W2A4KV4	TurboBoA	71.06	60.27	32.68	59.84	57.62	33.80	67.43	63.01	55.71
		KLIQ	72.03	61.83	34.04	60.19	59.19	35.40	68.84	64.78	**57.04**

Note: All quantization methods use the OSTQuant transformation.

E.2 Comparison with GPTAQ and BoA

We provide additional experimental results comparing KLIQ with GPTAQ (Li et al., 2025) and BoA (Kim et al., 2025). BoA introduces attention-aware Hessians but quantizes one output channel at a time, without adaptive grid search, coordinate-descent (CD) scale refinement, or compensation for errors propagated from previously quantized layers. TurboBoA extends BoA with joint quantization of multiple output channels and these additional mechanisms. GPTAQ compensates for accumulated errors through asymmetric calibration against full-precision layer outputs, using an input Gram matrix as its column Hessian factor and an identity row factor. In contrast, KLIQ derives its Q/K curvature factors from an attention-distribution KL objective, incorporating attention Fisher information into the query row factor and key column factor, with nontrivial row factors that capture output-channel dependencies. These factors and the corresponding propagated-error statistics guide quantization and compensation.

Table 7 compares these methods under 2-bit weight-only quantization across ten models. All methods use QuaRot, and both KLIQ and BoA incorporate the RMSNorm-aware variant for Qwen3. KLIQ achieves the best zero-shot average accuracy on eight of ten models and the best five-shot MMLU accuracy on nine. Averaged across models, it improves zero-shot accuracy by 2.69 and 1.46 percentage points over GPTAQ and BoA, respectively, with corresponding MMLU gains of 3.16 and 1.89 points. Against BoA, KLIQ also consistently reduces perplexity on both Wiki2 and C4 across all ten models.

Although GPTAQ retains a perplexity advantage on several models, this does not consistently translate into higher downstream accuracy. On Qwen3-8B and Mistral-7B, KLIQ achieves better zero-shot and MMLU accuracy despite higher perplexity. Gains are less consistent on the smallest models, but the overall results demonstrate broad downstream benefits across model families, supporting the effectiveness of KLIQ under aggressive quantization.

Table 7: Additional 2-bit weight-only quantization results

(a) PPL (↓)

Method	LLaMA3.2-1B		LLaMA3.2-3B		LLaMA3-8B		LLaMA2-7B		LLaMA2-13B	
	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4
GPTAQ	**45.11**	**45.55**	27.69	30.24	27.71	24.47	15.13	16.50	11.06	12.67
BoA	49.92	59.87	28.90	34.32	25.60	30.02	14.20	16.55	10.60	12.85
KLIQ	49.86	53.30	**25.99**	**29.04**	**21.87**	**24.97**	**12.72**	**14.43**	**10.00**	**11.79**
Method	**Qwen3-1.7B**		**Qwen3-4B**		**Qwen3-8B**		**Qwen3-14B**		**Mistral-7B-v0.3**	
	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4	Wiki2	C4
GPTAQ	**31.29**	**35.10**	23.39	28.57	**14.53**	19.56	14.88	19.51	**10.89**	**13.56**
BoA	73.76	80.35	30.68	37.57	17.22	22.65	13.62	19.03	14.00	17.79
KLIQ	44.68	46.11	**21.47**	**26.81**	14.97	**20.36**	**12.42**	**17.49**	11.90	14.98

(b) Zero-shot accuracy (↑) and five-shot MMLU (↑)

Method	LLaMA3.2-1B		LLaMA3.2-3B		LLaMA3-8B		LLaMA2-7B		LLaMA2-13B	
	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU
GPTAQ	39.04	25.44	43.45	25.71	50.70	**28.59**	49.29	28.00	55.92	32.07
BoA	**39.47**	25.20	46.62	27.32	52.05	26.83	53.40	28.34	58.64	35.64
KLIQ	38.79	**25.98**	**47.78**	**27.77**	**53.00**	28.50	**53.97**	**28.89**	**59.47**	**36.38**
Method	**Qwen3-1.7B**		**Qwen3-4B**		**Qwen3-8B**		**Qwen3-14B**		**Mistral-7B-v0.3**	
	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU	Avg.	MMLU
GPTAQ	**44.21**	27.74	50.33	35.58	55.03	44.24	56.21	44.94	56.40	33.13
BoA	38.93	25.17	48.96	36.52	57.00	46.67	61.03	57.96	56.77	28.45
KLIQ	42.34	**28.56**	**52.55**	**39.02**	**58.08**	**48.28**	**63.70**	**58.91**	**57.76**	**34.75**

Note: All quantization methods use the QuaRot transformation. KLIQ and BoA use the RMSNorm-aware variant for Qwen3 (Base) models.

E.3 Memory and Runtime Efficiency

Table 8 reports peak allocated GPU memory and quantization runtime for KLIQ and TurboBoA on five LLaMA models. KLIQ requires 1.5-11.7% more peak memory, with an average increase of 5.9% across models. The additional memory is particularly small on the LLaMA3 family, ranging from 0.30 to 1.07 GB. Runtime increases by 22.6-45.7%, averaging 30.5%, with the smallest relative overhead observed on LLaMA2-13B.

These costs accompany consistent quality improvements in Section 4.2. On the same five LLaMA models, KLIQ outperforms TurboBoA on all four aggregate metrics in Table 1, reducing Wiki2 and C4 perplexity by an average of 4.48% and 3.14%, respectively, and improving zero-shot average accuracy and five-shot MMLU by 0.44 and 1.15 percentage points. For example, on LLaMA3-8B, KLIQ improves zero-shot accuracy by 0.83 points and MMLU by 2.33 points, while requiring only 1.07 GB of additional peak memory and 19.08 minutes of additional quantization time.

Overall, these measurements establish a favorable quality-cost trade-off: KLIQ delivers consistent improvements under aggressive 2-bit quantization with a limited increase in peak memory and additional offline processing time. The results complement the main evaluation by quantifying the preparation cost associated with KLIQ's accuracy gains.

Table 8: Memory and runtime on LLaMA2 and LLaMA3

Model	KLIQ		TurboBoA	
	Memory (GB)	Time (min)	Memory (GB)	Time (min)
LLaMA 3.2-1B	19.98	13.02	19.68	8.935
LLaMA 3.2-3B	21.22	37.24	20.59	28.35
LLaMA 3-8B	40.98	90.74	39.91	71.66
LLaMA 2-7B	42.54	84.48	38.47	66.87
LLaMA 2-13B	74.97	183.6	67.09	149.8

Note: Memory denotes peak allocated GPU memory in GB.