PROJECTED-DISTORTION GUIDED QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training quantization (PTQ) reduces the memory and latency costs of large language models (LLMs), but existing methods degrade sharply at 2–3 bits, in part because codebook fitting, outlier handling, and pruning are designed and tuned separately. We present *Projected-Distortion Guided Quantization* (PDGQ), which drives all three with a single activation-aware measure of output distortion. PDGQ combines (1) a maximum-entropy codebook refined by activation-weighted Lloyd iterations, (2) Projected-Distortion Outlier Restoration (PDOR), which restores weights whose distortion is large relative to a rate–distortion reference, on a scale shared across layers and bit-widths, and (3) Distortion-Consistent Pruning (DCP), which zeroes weights that zero represents better than their codeword. On LLaMA and Qwen3 models, PDGQ matches recent PTQ methods in perplexity and zero-shot accuracy at 4 bits and outperforms them by a margin that grows as the bit budget shrinks, being largest at nominal 2–3 bits, with a few narrow exceptions that we report. With simple activation and KV-cache quantization, PDGQ also preserves DeepSeek-R1-Distill accuracy on GSM8K and MATH-500 under W4A4KV4 better than FlatQuant and QuaRot.

1 INTRODUCTION

Large Language Models (LLMs) have redefined the frontier of artificial intelligence, delivering advances in natural language understanding, generation, multimodal reasoning, and scientific discovery Grattafiori et al. (2024). Deployment, however, is constrained by memory footprint and inference latency: a 70B-parameter model in FP16 requires roughly 140 GB of GPU memory Grattafiori et al. (2024); Lin et al. (2024), well beyond commodity hardware. As models move toward native multimodality and longer contexts, this gap widens, creating a need for compression that is near-lossless and requires neither retraining nor architectural changes.

Post-training quantization (PTQ) addresses this need, but at 3 bits and especially 2 bits the quality of scalar quantizers drops sharply. A contributing reason is that a modern PTQ pipeline is assembled from components designed in isolation. Codebooks or scales are fitted to a (Hessian- or activation-) weighted reconstruction loss Frantar et al. (2023); Lin et al. (2024); Kim et al. (2023), outliers are selected by a magnitude percentile or a sensitivity threshold Kim et al. (2023); Dettmers et al. (2024), and sparsity is set by an externally chosen target Frantar & Alistarh (2023). Each component has its own threshold on its own scale, and none of them asks the question that matters at a fixed bit budget, namely how far a particular weight's error is from what b bits can achieve. Vector-quantization methods Tseng et al. (2024a); Egiazarian et al. (2024); Tseng et al. (2024b) close much of the low-bit gap by approaching the rate–distortion limit directly, but at the cost of substantially more complex decoding than a scalar codebook lookup.

We introduce *Projected-Distortion Guided Quantization* (PDGQ), a scalar PTQ method in which a single activation-weighted distortion measure drives codebook construction, outlier restoration, and pruning, and

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.

in which the restoration decision is made on a scale normalized by the rate–distortion floor $D^*(b)$. Our contributions are as follows.

- **Projected-Distortion Outlier Restoration (PDOR).** A per-weight statistic that measures a weight's contribution to output distortion relative to the floor-level contribution of its group. Because it is normalized by both the weight variance and the activation scale of the group, a single significance level α applies across layers, groups, and bit-widths.
- **Entropy-regularized, activation-weighted codebooks.** A maximum-entropy quantile initialization followed by activation-weighted Lloyd refinement with an entropy penalty, so that codewords are placed by output impact while their usage stays close to uniform.
- **Distortion-Consistent Pruning (DCP).** A parameter-free rule that zeroes a weight exactly when zero is closer to it than its codeword under the same distortion measure, and which never overrides a restoration decision.

2 Related Work

Scalar PTQ. Most PTQ pipelines quantize one layer or block at a time from a small calibration set, a paradigm established by GPTQ Frantar et al. (2023), which minimizes a Hessian-weighted reconstruction error with column-wise error compensation. Later methods rescale salient input channels (AWQ Lin et al. (2024), Astro Chen et al. (2026), PT2-LLM Yan et al. (2026)), shrink weight magnitudes (MagR Zhang et al. (2024)), learn clipping and equivalent transformations (OmniQuant Shao et al. (2024)) or rounding (AdaRound Nagel et al. (2020)), align layer outputs in closed form (LoaQ Lin & Wan (2025)), select a PTQ algorithm per layer (CALM Zhang et al. (2025)), or adapt parameters per group (GWQ Yang et al. (2024), GroupQ Jiang et al. (2024), GQSA Zeng et al. (2025), M-ANT Hu et al. (2025), TSGQ Kim et al. (2026), SliderQuant Wang et al. (2026)). These methods work well at 4 bits and above, but they optimize an aggregate reconstruction loss without reference to what the bit-width can achieve, and they degrade substantially at 2–3 bits.

Codebooks, outliers, and pruning. SqueezeLLM Kim et al. (2023) fits non-uniform codebooks by sensitivity-weighted k-means and keeps a fixed fraction of large-magnitude weights in full precision; SpQR Dettmers et al. (2024) keeps weights whose second-order sensitivity exceeds a hand-set threshold; OWQ Lee et al. (2024), OS+ Wei et al. (2023), and QLLM Liu et al. (2024a) target heavy-tailed weights or activations; and SparseGPT Frantar & Alistarh (2023) prunes to a chosen sparsity. PDGQ's codebook is closest to SqueezeLLM's sensitivity-weighted k-means, with activation energy in place of the Fisher information and an entropy term, and its restoration step is closest to SpQR's, but uses a threshold on a floor-normalized scale rather than one set per model.

Vector quantization and information-theoretic PTQ. QuIP# Tseng et al. (2024a), AQLM Egiazarian et al. (2024), GPTVQ van Baalen et al. (2024), and QTIP Tseng et al. (2024b) quantize weights jointly with lattice, additive, vector, or trellis codebooks, approaching the rate–distortion bound and defining the state of the art at 2 bits, at the cost of more complex decoding. WaterSIC Lifar et al. (2026), EWQ Behtash et al. (2025), LieQ Xiao et al. (2025), and Radio Young (2025) allocate bits across layers or groups using entropy or mutual-information criteria, while Q-Palette Lee & Song (2025) and NestQuant Savkin et al. (2025) design flexible or lattice-structured codebooks. PDGQ is complementary: it keeps a scalar codebook within a fixed bit allocation and applies a rate–distortion reference to individual weights.

3 PRELIMINARIES

Let $W \in \mathbb{R}^{O\times I}$ be the weight matrix of a linear layer, $\hat{W}$ its quantized counterpart, $E = W - \hat{W}$ the error, and $\mathbf{x} \in \mathbb{R}^I$ the input activation.

Projected distortion. We measure quantization error by the expected squared output error,

$$\mathcal{L}(W, \hat{W}) = \mathbb{E}_{\mathbf{x}}\big[\|E\mathbf{x}\|_2^2\big] = \mathrm{Tr}\big(E\,\Sigma_x E^\top\big), \qquad \Sigma_x = \mathbb{E}[\mathbf{x}\mathbf{x}^\top], \tag{1}$$

which reduces to weight-space distortion $\|E\|_F^2$ when $\Sigma_x = \mathbf{I}$ and otherwise emphasizes errors along strongly activated input directions. Keeping only the diagonal of Σ_x gives the per-weight decomposition

$$\mathcal{L}(W, \hat{W}) \;\approx\; \sum_{o,i} e_{oi}^2\, \sigma_x^2(i), \qquad \sigma_x^2(i) = \mathbb{E}[x_i^2] - \mathbb{E}[x_i]^2, \tag{2}$$

where $e_{oi} = w_{oi} - \hat{w}_{oi}$, and $\sigma_x^2(i)$, the *activation energy* of column i, is the per-column variance of calibration activations $\mathbf{X} \in \mathbb{R}^{I\times N}$. The decomposition drops the cross terms, which vanish in expectation when errors are uncorrelated across columns, and centers each column, which places $\sigma_x^2(i)$ below $[\Sigma_x]_{ii}$ by $\mathbb{E}[x_i]^2$. In exchange, each weight's contribution can be evaluated independently, as the decisions of Sections 4.2–4.3 require and full second-order methods such as GPTQ Frantar et al. (2023) give up. On real layers the diagonal of Σ_x itself recovers the exact distortion to within 1% at the median, the variance sits about 10% below it, and the variance yields perplexity no worse than the uncentered choice.

Rate–distortion reference. Modelling the weights of a row-group as i.i.d. Gaussian with variance σ_w^2, the rate–distortion function $R(D) = \frac{1}{2}\log_2(\sigma_w^2/D)$ Shannon et al. (1959) implies that no quantizer at b bits per weight achieves MSE below

$$D^*(b) = \sigma_w^2\, 2^{-2b}. \tag{3}$$

Because the Gaussian is the hardest source of a given variance to compress, $D^*(b)$ also upper-bounds the optimal distortion of any source with variance σ_w^2. The bound is attained only by vector quantization over long blocks; the best fixed-rate scalar quantizer with $K = 2^b$ codewords (Lloyd–Max) stays $1.88\times$ above it at $b = 2$ and $2.21\times$ at $b = 3$. We therefore use $D^*(b)$ as a *reference scale* for per-weight error, not as a target, and require the quantizer both to minimize distortion and to use its K codewords fully.

4 METHODOLOGY

$D^*(b)$ describes average distortion but not how a real quantizer's error is distributed across weights. Figure 1 shows this distribution for one layer through the statistic T_{oi} of Section 4.2: a weight's projected-distortion contribution divided by the floor-level contribution of its group. Most weights sit near the reference level, $T_{oi} \approx 1$, but the distribution has a long right tail: 9.3% of weights at $b = 2$ and 11.2% at $b = 3$ exceed the $\chi^2_{1-\alpha}(1)$ cutoff, some by two orders of magnitude, and together they carry 53% ($b = 2$) and 66% ($b = 3$) of the layer's projected distortion. A further, disjoint 16.8% of weights (6.9% at $b = 3$) lie closer to zero than to their codeword. Across all decoder layers of the model, 24.3% of weights fall into one of these two populations at $b = 2$ and 16.7% at $b = 3$. In addition, Lloyd refinement without an entropy term leaves 5.7% ($b = 2$) and 3.2% ($b = 3$) of the b bits of codeword entropy unused.

4.1 PROJECTED-DISTORTION FOR CODEBOOK OPTIMIZATION

Each row-group $\mathbf{w}_{o,g} = \{w_{oi}\}_{i\in g}$ receives its own codebook $\mathbf{c} = (c_1, \ldots, c_K)$, $K = 2^b$, and assignments $a_{oi} \in \{1, \ldots, K\}$, with quantized value $q_{oi} = c_{a_{oi}}$.

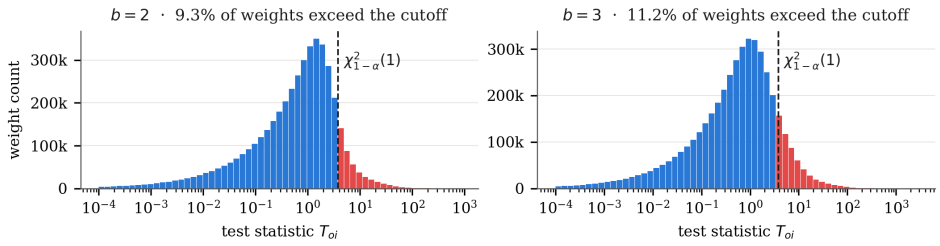


Figure 1: Distribution of the PDOR statistic T_{oi} (log scale) for the MLP down-projection of layer 10 of Qwen2.5-0.5B-Instruct ($b = 2$ left, $b = 3$ right; group size 128, $\alpha = 0.05$). Most weights lie near the reference level $T_{oi} \approx 1$. The dashed line marks the $\chi^2_{1-\alpha}(1)$ cutoff; the weights beyond it (red; 9.3%/11.2%) are restored by PDOR and carry 53%/66% of the layer's projected distortion.

Why entropy. Because quantization is deterministic, $H(\hat{W} \mid W) = 0$, so $I(W; \hat{W}) = H(\hat{W})$: the information the quantized weights retain about the originals is exactly the entropy of codeword usage. This is at most $\log_2 K = b$ bits, attained when all codewords are used equally often. A codebook that minimizes distortion alone may leave codewords nearly unused, which wastes part of the fixed b-bit budget.

Objective. We therefore minimize projected distortion with an entropy regularizer:

$$\min_{\mathbf{c},\,\mathbf{a}} \underbrace{\sum_{i \in g} \sigma_x^2(i)\,(w_{oi} - c_{a_{oi}})^2}_{\text{projected distortion}} - \gamma\,|g| \underbrace{H(\mathbf{p})}_{\text{usage entropy}}, \qquad p_k = \tfrac{1}{|g|}\big|\{i \in g : a_{oi} = k\}\big|, \tag{4}$$

where H is measured in nats and $\gamma \geq 0$ trades distortion against codeword utilization. The factor $|g|$ puts both terms on a per-weight scale. We optimize Eq. 4 in two stages.

Maximum-entropy initialization. Rather than splitting the value range into equal-width bins, which wastes codewords on sparse tails, we place bin boundaries at empirical quantiles,

$$\mathcal{B}_k = \mathrm{Quantile}\Big(\mathbf{w}_{o,g}, \frac{k}{K}\Big), \quad k = 1, \ldots, K-1, \tag{5}$$

so that $p_k \approx 1/K$ and $H(\mathbf{p})$ is at its maximum at initialization.

Activation-weighted Lloyd refinement. Starting from this assignment, we alternate for J iterations:

- **Assignment:** $a_{oi} = \arg\min_k \big[\sigma_x^2(i)\,(w_{oi} - c_k)^2 + \gamma \ln p_k\big]$,
- **Update:** $c_k = \dfrac{\sum_{i \in g:\, a_{oi} = k} \sigma_x^2(i)\, w_{oi}}{\sum_{i \in g:\, a_{oi} = k} \sigma_x^2(i)}$,

with p_k recomputed after each assignment step. The update is the exact minimizer of Eq. 4 for fixed assignments. The assignment term $\gamma \ln p_k$ is the first-order change in $-\gamma|g|H(\mathbf{p})$ when one weight joins codeword k; it favours under-used codewords and so counteracts the collapse in usage that plain Lloyd iterations produce. Since this step is a linearization, the objective is not guaranteed to decrease monotonically; in practice it converges within $J \approx 12$ iterations.

4.2 PROJECTED-DISTORTION OUTLIER RESTORATION (PDOR)

Even a well-fitted codebook leaves some weights with a large contribution to Eq. 2, either because their residual is large or because they sit on a high-energy input column. PDOR identifies these weights by comparing each contribution with a common reference, the floor-level contribution of its group, rather than with a magnitude percentile or a per-model threshold.

Statistic. For row-group (o, g), let $D^*_{o,g}(b) = \hat{\sigma}^2_{w,o,g}\, 2^{-2b}$, where $\hat{\sigma}^2_{w,o,g}$ is the sample variance of $\mathbf{w}_{o,g}$, and let $\bar{\sigma}^2_x(g) = \frac{1}{|g|}\sum_{j\in g}\sigma^2_x(j)$ be the mean activation energy of the group's input columns. We define

$$T_{oi} = \frac{e^2_{oi}\,\sigma^2_x(i)}{D^*_{o,g}(b)\,\bar{\sigma}^2_x(g)} = \underbrace{\frac{e^2_{oi}}{D^*_{o,g}(b)}}_{\text{error relative to floor}} \cdot \underbrace{\frac{\sigma^2_x(i)}{\bar{\sigma}^2_x(g)}}_{\text{relative importance}}, \qquad e_{oi} = w_{oi} - q_{oi}. \tag{6}$$

T_{oi} is the weight's projected-distortion contribution divided by the contribution that an average-importance weight in the same group would make if its error were at the rate–distortion floor. It is dimensionless and invariant to rescaling the weights or the activations of a group, so the same threshold applies to every layer, group, and bit-width.

Threshold. As a reference model we take the residual of an ideal quantizer operating at the floor, $e_{oi} \sim \mathcal{N}(0, D^*_{o,g}(b))$ (the backward test channel of the Gaussian rate–distortion problem; Appendix D.3.11). Under this reference, the first factor of Eq. 6 is $\chi^2(1)$-distributed, and we restore

$$\hat{w}_{oi} \;\leftarrow\; w_{oi} \qquad \text{if } T_{oi} > \chi^2_{1-\alpha}(1). \tag{7}$$

This reference is a scale, not a model of the actual residuals. A scalar quantizer operates above the floor (Section 3), so the fraction of restored weights $\rho_{\mathcal{R}}$ is not α: for a Lloyd–Max quantizer on Gaussian weights and uniform importance it would be about 8% at $b = 2$ and 13% at $b = 3$ for $\alpha = 0.05$, whereas the realized fraction depends on the codebook and on the importance factor below and is about 1% for LLaMA-2-7B. Accordingly, we treat α as a single knob on the restoration budget and report the realized $\rho_{\mathcal{R}}$ and its storage cost in the component analysis. The importance factor lowers the effective threshold for high-energy columns and raises it for low-energy ones, so the restored set differs from the largest-magnitude weights: at $b = 2$, only 46% of PDOR-restored weights are among the top-$\rho_{\mathcal{R}}$ weights by magnitude (Qwen2.5-0.5B-Instruct, $\alpha = 0.05$).

Restoration replaces q_{oi} with the original value, removing e_{oi} at the cost of storing one full-precision value and its position.

4.3 DISTORTION-CONSISTENT PRUNING (DCP)

A scalar codebook with an even number of levels typically has no codeword at zero, so a weight near zero incurs the full distance to the nearest inner codeword. For such weights, zero is a better representation than the codeword.

Pruning criterion. Under Eq. 2, keeping q_{oi} and replacing it with zero cost

$$d_{\text{keep}} = \sigma^2_x(i)\, e^2_{oi}, \qquad d_{\text{prune}} = \sigma^2_x(i)\, w^2_{oi}. \tag{8}$$

Both terms carry the same activation energy, so the comparison reduces to

$$d_{\text{prune}} \le d_{\text{keep}} \iff |w_{oi}| \le |w_{oi} - q_{oi}|, \tag{9}$$

i.e. zero is at least as close to w_{oi} as its codeword. DCP is therefore nearest-neighbour quantization over the augmented set $\{c_1, \ldots, c_K\} \cup \{0\}$. It introduces no threshold or sparsity target; the number of pruned weights follows from the codebook.

Interaction with PDOR. Weights in the restored set $\mathcal{R}$ are exempt. A restored weight has zero error, so pruning it can only increase distortion. The two rules are complementary: PDOR keeps weights whose quantization error is too costly, and DCP removes codewords that do worse than zero.

Final weight assignment.

$$\hat{w}_{oi} = \begin{cases} w_{oi}, & (o,i) \in \mathcal{R}, \\ 0, & (o,i) \notin \mathcal{R} \text{ and } |w_{oi}| \leq |w_{oi} - q_{oi}|, \\ q_{oi}, & \text{otherwise.} \end{cases} \tag{10}$$

The result is a mixed representation of full-precision values, zeros, and codewords.

5 EXPERIMENT

5.1 SETUP

We assess the impact of quantization using perplexity (WikiText2, C4), zero-shot downstream reasoning via the lm-evaluation-harness Biderman et al. (2024) on six standard benchmarks, and open-ended math reasoning (GSM8K, MATH-500). Full dataset and benchmark descriptions are provided in Appendix B.

We evaluate PDGQ on the LLaMA family (7B–70B) against a comprehensive suite of state-of-the-art PTQ baselines spanning magnitude-based, activation-aware, information-theoretic, and structural approaches (surveyed in Appendix A), followed by further comparison using Qwen3-Base family and on DeepSeek-R1-Distill-Qwen reasoning models. Baseline results are quoted from the original publications where available, and missing entries were re-run under the same protocol. Full experimental configuration, hyperparameters, and hardware/software environment are in Appendix C.

5.2 LANGUAGE MODELING

#bits	Method	1-7B	1-13B	1-30B	1-65B	2-7B	2-13B	2-70B	3-8B	3-70B
16	-	5.68	5.09	4.10	3.53	5.47	4.88	3.31	5.75	2.90
4-bit	CBQ Ding et al. (2025)	5.72	5.14	4.14	3.59	**5.52**	4.94	3.42	6.88	3.10
	SliderQuant Wang et al. (2026)	5.71	5.11	4.14	3.58	5.61	5.00	3.41	6.79	3.06
	Radio Young (2025)	5.73	5.13	4.15	3.60	5.57	4.97	3.40	6.91	3.12
	NestQuant Savkin et al. (2025)	**5.70**	**5.10**	4.15	3.60	5.53	4.93	3.38	6.31	3.14
	LRQ Lee et al. (2025)	5.74	5.12	4.16	3.61	5.75	5.02	3.47	6.94	3.20
	PDGQ	5.72	5.12	**4.13**	**3.56**	**5.52**	**4.90**	**3.34**	**6.23**	**2.94**
3-bit	SliM-LLM Huang et al. (2025)	6.40	5.48	4.61	3.99	6.24	5.26	3.67	7.16	4.08
	CBQ Ding et al. (2025)	6.42	5.50	4.63	4.01	5.89	5.23	3.69	7.18	4.02
	Radio Young (2025)	6.45	5.52	4.65	4.03	6.04	5.25	3.72	7.30	4.08
	LRQ Lee et al. (2025)	6.46	5.54	4.67	4.05	6.48	5.57	3.89	7.12	4.12
	PDGQ	**5.75**	**5.25**	**4.24**	**3.68**	**5.65**	**5.02**	**3.44**	**6.55**	**3.33**
2-bit	SliM-LLM Huang et al. (2025)	14.58	8.87	7.33	5.90	16.01	9.41	6.28	39.66	9.46
	CBQ Ding et al. (2025)	10.60	7.50	5.58	5.25	8.01	7.00	5.10	26.80	8.15
	SliderQuant Wang et al. (2026)	10.30	7.40	5.65	5.15	9.59	7.71	6.53	27.59	8.10
	PDGQ	**6.57**	**5.84**	**4.83**	**4.18**	**6.50**	**5.54**	**3.94**	**7.94**	**4.98**

Table 1: WikiText2 perplexity against the most recent (2025+) PTQ baselines (bold and underline mark best and second best). Full comparison in Appendix G.

LLaMA Family We compare the perplexity results on WikiText2 and C4 across multiple LLaMA model families and bit-widths. Overall, PDGQ consistently achieves state-of-the-art or highly competitive performance, with the largest gains appearing in the ultra-low-bit regime. Comprehensive numerical results for both WikiText2 and C4 are provided in Appendix G.

PDGQ matches or outperforms baselines such as AWQ, GPTQ, CBQ, and SliderQuant already at 4-bit (e.g. 3.56/3.34/2.94 WikiText2 on LLaMA-1 65B/LLaMA-2 70B/LLaMA-3 70B), trailing NestQuant by 0.02 on LLaMA-1 7B and 13B, and the gap widens sharply as bit-width drops. At 3-bit, where existing methods begin to degrade rapidly, PDGQ stays stable, cutting LLaMA-3 70B perplexity from 4.08 to 3.33 versus SliM-LLM. The largest gains appear at 2-bit, where RTN and AWQ become unstable and even SliM-LLM degrades sharply: PDGQ still reaches 3.94 on LLaMA-2 70B and 7.94 on LLaMA-3 8B (vs. SliM-LLM's 39.66). C4 shows a similar pattern (e.g. 6.31 vs. 7.01/9.59 for SliM-LLM/SliderQuant on LLaMA-2 70B at 2-bit), except that Radio is ahead of PDGQ on LLaMA-2 at 2 bits (5.99 vs. 6.31 on 70B). These results hold across model families, scales, and both datasets, with that exception. Table 1 reports WikiText2 perplexity against the most recent (2025+) PTQ baselines; the full comparison, including earlier methods (RTN, GPTQ, QuIP, AWQ, PB-LLM), is in Appendix G. Next, we extend these gains to Qwen3-Base (0.6B–14B) and isolates PDGQ's codebook and outlier criterion against QuIP and SqueezeLLM.

Qwen3-Base. Table 2 extends the comparison to the Qwen3-Base family (0.6B–14B) against five baselines on WikiText2 and C4. At 4-bit, GPTQ is marginally ahead of PDGQ on most sizes (a tie at 6.7 on 14B WikiText2 and a $\leq$0.3-point edge elsewhere), while PDGQ beats SliderQuant and Radio at every size and dataset. At 3 and 2 bits PDGQ is best in all 20 cells, for example 17.5 vs. 20.0/21.5 (SliderQuant/Radio) at 3 bits and 30.5 vs. 40.0/46.0 at 2 bits on 0.6B WikiText2, while AWQ and GPTQ collapse at 2 bits. As on LLaMA, the advantage is negligible at 4-bit and decisive in the ultra-low-bit regime (Appendix I for details).

Bits	Method	0.6B		1.7B		4B		8B		14B	
		WT2	C4	WT2	C4	WT2	C4	WT2	C4	WT2	C4
16	fp16	12.7	17.1	9.4	13.4	7.9	11.6	7.0	10.4	6.4	9.7
4-bit	AWQ Lin et al. (2024)	16.6	21.3	11.4	15.8	9.4	13.3	8.1	11.6	7.3	10.7
	GPTQ Frantar et al. (2023)	14.9	19.7	10.0	**14.0**	**8.2**	**11.9**	**7.2**	**10.6**	**6.7**	**9.9**
	SliderQuant Wang et al. (2026)	15.0	20.0	10.4	14.7	8.6	12.6	7.5	11.0	6.9	10.3
	Radio Young (2025)	15.3	20.3	10.6	15.0	8.8	12.8	7.6	11.1	7.0	10.4
	PDGQ	**13.8**	**18.5**	**9.9**	14.2	8.3	12.2	7.3	10.9	**6.7**	10.2
3-bit	AWQ Lin et al. (2024)	85.9	89.3	41.8	40.4	26.3	26.4	22.6	23.8	19.2	23.1
	SliderQuant Wang et al. (2026)	20.0	26.0	13.8	18.3	11.3	15.4	10.4	13.7	9.1	12.4
	Radio Young (2025)	21.5	27.5	14.6	19.1	11.9	16.0	10.7	14.0	9.5	12.8
	PDGQ	**17.5**	**23.0**	**12.8**	**17.2**	**10.9**	**14.8**	**10.1**	**13.3**	**8.8**	**12.0**
2-bit	AWQ Lin et al. (2024)	6.4e7	9.8e7	1.1e7	1.0e7	7.5e6	5.9e6	1.7e7	1.3e7	2.7e7	2.2e7
	GPTQ Frantar et al. (2023)	7.5e3	3.8e3	2.2e2	1.5e2	1.1e2	79.0	53.1	36.6	27.9	24.5
	SliderQuant Wang et al. (2026)	40.0	48.5	24.5	30.0	18.0	22.5	14.7	18.7	12.3	16.2
	Radio Young (2025)	46.0	55.0	27.0	33.0	19.5	24.5	15.8	19.8	13.0	17.0
	PDGQ	**30.5**	**36.5**	**19.8**	**24.0**	**15.6**	**19.5**	**13.4**	**17.0**	**11.6**	**15.8**

Table 2: WikiText2 (WT2) and C4 perplexity across the Qwen3-Base family. **Bold** marks the best of the reported quantized methods per column.

5.3 Zero-shot Reasoning

Zero-shot results show that PDGQ preserves downstream reasoning under aggressive compression, achieving the best or near-best accuracy on most benchmarks across the LLaMA families. The advantage is largest at 2 bits: on LLaMA-3 70B, PDGQ retains 96–99% of FP16 accuracy on PIQA, HellaSwag, WinoGrande,

and ARC-E (94% on ARC-C), and on LLaMA-1 30B it raises HellaSwag to 79.39 versus 71.55 for QuIP and 66.10 for SliM-LLM. At 3 and 4 bits the gap narrows, but PDGQ remains competitive or best, particularly on reasoning-heavy tasks. That the margin widens as bit-width decreases is consistent with our premise that per-weight, floor-referenced error allocation matters most when the bit budget is tightest. Details are in Appendix H.

5.4 Math Reasoning under WxAxKVx Quantization

To test whether PDGQ's benefits extend to activation and KV-cache compression, we additionally apply simple per-channel round-to-nearest scaling to activations and the KV-cache, independent of the weight codebook, and evaluate DeepSeek-R1-Distill-Qwen at 1.5B, 7B, and 14B on GSM8K Cobbe et al. (2021) and MATH-500 Lightman et al. (2024) under the exact protocol of Table 1 in Liu et al. (2025) (32768 max generated tokens, temperature 0.6, top-p 0.95; full quantization and decoding configuration in Appendix C). BF16 and FlatQuant/QuaRot numbers for 7B and 14B are taken directly from Table 1 of Liu et al. (2025); QuaRot is not reported for 7B in that paper (skipped by its authors due to an incompatible hidden dimension for the Hadamard transform).

Config	Method	1.5B		7B		14B	
		MATH-500	GSM8K	MATH-500	GSM8K	MATH-500	GSM8K
BF16	-	84.40	84.61	94.60	91.43	94.60	93.86
W8A8KV8	FlatQuant Sun et al. (2025)	84.80	85.06	93.00	91.13	**95.20**	94.01
	QuaRot Ashkboos et al. (2024)	85.20	83.32	–	–	93.80	**94.24**
	PDGQ	**86.27**	**86.79**	**94.20**	**91.68**	95.00	94.18
W4A4KV4	FlatQuant Sun et al. (2025)	64.80	78.62	82.80	90.83	93.40	93.40
	QuaRot Ashkboos et al. (2024)	1.20	0.76	–	–	91.20	93.18
	PDGQ	**69.80**	**80.65**	**86.90**	**91.35**	**93.70**	**93.75**

Table 3: GSM8K / MATH-500 accuracy for DeepSeek-R1-Distill-Qwen at 1.5B, 7B, and 14B. BF16 and FlatQuant/QuaRot numbers are from Table 1 of Liu et al. (2025); "–" marks a configuration not reported there (QuaRot is skipped for 7B). **Bold** marks the best of the three quantized methods per column.

PDGQ wins 10 of the 12 quantized-method comparisons. The two exceptions are both at W8A8KV8 on the 14B model, where it trails FlatQuant on MATH-500 (95.00 vs. 95.20) and QuaRot on GSM8K (94.18 vs. 94.24) by less than a third of a point, effectively a tie when all three methods are within 1.5 points of BF16. At the more aggressive W4A4KV4 setting PDGQ leads every method on every model and benchmark, by wide margins over FlatQuant on MATH-500 at 1.5B (69.80 vs. 64.80) and 7B (86.90 vs. 82.80), while QuaRot collapses to near-random accuracy at 1.5B and is not applicable at 7B. As with perplexity, the advantage grows as the bit budget shrinks, here even though activations and the KV-cache are quantized with only a simple RTN scheme.

5.5 Analysis

We analyse the contribution of each PDGQ component on LLaMA-2-7B and Qwen3-8B-Base (WikiText2 perplexity, group size 128 unless stated). Full tables are in Appendices J–L. Bit-widths below are the nominal codebook bit-widths; effective bits per weight, including restored weights and pruning masks, are given in Appendix F.

The activation-weighted codebook gives the largest single gain. Replacing unweighted Lloyd refinement (same quantile initialization) with the activation-weighted codebook of Section 4.1 lowers LLaMA-2-7B perplexity from 10.58 to 8.68 at 2 bits and from 7.13 to 6.13 at 3 bits, at no change in bit-width. This

is the largest improvement of any component at every bit-width. On its own, the codebook already reaches lower perplexity than QuIP Chee et al. (2023) at 2, 3, and 4 bits (8.68 vs. 27.13 at 2 bits; Appendix K).

Maximum-entropy initialization accelerates refinement. Starting from quantile boundaries, perplexity before any Lloyd iteration is already 5.8 at 3 bits and 5.6 at 4 bits, compared with 9.2 and 6.1 from random initialization, and refinement converges within about five iterations (Appendix L). With this initialization, 2-bit codebooks improve up to $J \approx 12$ iterations and 3- and 4-bit codebooks plateau by $J \approx 8$–10; beyond $J = 16$ gains are negligible (Appendix L). We use $J = 20$ throughout.

Output distortion is concentrated in a small set of weights. Restoring 0.93% of weights with PDOR lowers 2-bit perplexity from 8.68 to 7.55 at a cost of 0.20 bits, closing 36% of the gap to the 4-bit codebook. Per extra bit, this is 5.7 perplexity points, more than twice the 2.6 points gained by raising the whole codebook from 2 to 3 bits; at 3 bits the ratio is similar (1.2 vs. 0.6). Weighting the statistic by activation energy restores fewer weights than the unweighted test (0.93% vs. 1.13% at 2 bits) with slightly lower perplexity (7.55 vs. 7.60).

PDOR selects more useful weights than a magnitude rule. With the codebook and pruning settings fixed, replacing PDOR with a magnitude rule in the style of SqueezeLLM Kim et al. (2023) (restore weights outside mean $\pm\, 3\sigma$ of their row-group) increases perplexity in every setting, from 6.50 to 6.92 on LLaMA-2-7B and from 13.41 to 31.90 on Qwen3-8B-Base at 2 bits. On Qwen3 the magnitude rule ends above pruning alone with no restoration (31.90 vs. 28.64). PDOR also restores 2.4–5.7$\times$ more weights, so part of the gap reflects its larger budget.

DCP improves quality at a storage cost. DCP prunes 16.7%, 8.7%, and 3.8% of weights at 2, 3, and 4 bits and lowers 2-bit perplexity from 7.55 to 6.50. Because recording pruned positions costs more than the codes it replaces, it raises the effective bit-width (2.20 to 2.86 at 2 bits, 3.29 to 3.64 at 3 bits; Appendix F). Its gain per extra bit (1.6 perplexity points at 2 bits, 0.4 at 3 bits) is lower than PDOR's, and at 3 and 4 bits its effect is small (0.14 and 0.02). DCP's benefit is concentrated at 2 bits, where a codebook without a zero level misrepresents the largest share of near-zero weights.

The components are complementary and transfer across architectures. On Qwen3-8B-Base, PDOR and DCP each improve on the codebook alone at every bit-width, and combining them gives the lowest perplexity (13.41 vs. 32.37 at 2 bits). Their effects are approximately additive: at 2 bits, PDOR alone lowers perplexity by 15.3, DCP alone by 3.7, and both together by 19.0. As on LLaMA-2, PDOR contributes more than DCP at every bit-width.

6 Conclusions

We introduced *Projected-Distortion Guided Quantization* (PDGQ), a post-training quantization method grounded in rate–distortion theory in which one activation-aware distortion measure drives codebook construction, outlier restoration, and pruning, rather than three separately tuned heuristics. Across the LLaMA and Qwen3-Base families, and on GSM8K and MATH-500 for DeepSeek-R1-Distill-Qwen under joint weight, activation, and KV-cache quantization, PDGQ matches or exceeds state-of-the-art PTQ methods, with the largest gains at 2–3 bits, where magnitude-driven heuristics break down.

Limitations. PDGQ assumes Gaussian-like weights and a diagonal activation covariance, stores restored weights and pruning positions beyond the nominal bit-width, and is untested on mixture-of-experts models. Calibration sensitivity, multimodal models, cross-layer error interactions, and inference kernels also remain open (more detailed is in Appendix M).

AI USE STATEMENT

In this work, we used generative AI tools to help check a proof: the Gaussian-residual justification in Appendix D.3.11, to verify our analyses of the diagonal covariance approximation (Appendix D.2), and to help format equations, check grammars, typos, and improve the clarity and readability of the manuscript. We have not used generative AI tools to formulate the paper's main mathematical claims, develop the theoretical framework, propose hypotheses, design the main experiments, or implement the core quantization method; generating synthetic data and translation do not apply to this work. We have reviewed all AI-assisted work, checking the proof against our derivation and verifying every reported number against our own runs or the cited source. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses only publicly released pretrained language models and publicly available benchmark datasets (WikiText2, C4, PIQA, HellaSwag, WinoGrande, ARC-e/c, BoolQ, GSM8K, MATH-500), each accessed and used under its original license and access terms for research purposes only. It does not involve human subjects, collect new data, or release new datasets or models, and we are not aware of any direct ethical concerns raised by this work.

REPRODUCIBILITY STATEMENT

(This section is **recommended** and does not count toward the page limit.)

We describe the full PDGQ procedure as pseudocode in Appendix E, together with the hyperparameters, hardware, and calibration protocol used for every experiment in Appendix C and Appendix B. The Gaussian-residual assumption underlying the outlier-restoration test is stated and proved in Appendix D.3.11. We submit our implementation as supplementary material to support reproduction of all reported results.

REFERENCES

Selim An, Yeseong Kim, and Il hong Suh. Glowq: Group-shared LOw-rank approximation for quantized LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Alireza Behtash, Marijan Fofonjka, Ethan Baird, Tyler Mauer, Hossein Moghimifam, David Stout, and Joel Dennison. Universality of layer-level entropy-weighted quantization beyond model architecture and size. *arXiv preprint arXiv:2503.04704*, 2025.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, et al. Lessons from the trenches on reproducible evaluation of language models. *arXiv preprint arXiv:2405.14782*, 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*, 2020.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. Quip: 2-bit quantization of large language models with guarantees. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Ningning CHEN, Weicai Ye, and Ying Jiang. HBLLM: Wavelet-enhanced high-fidelity 1-bit quantization for LLMs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Xi Chen, Ming Li, Junxi Li, Changsheng Li, Peisong Wang, Lizhong Ding, Ye Yuan, and Guoren Wang. Astro: Activation-guided structured regularization for outlier-robust llm post-training quantization. *arXiv preprint arXiv:2602.07596*, 2026.

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.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tim Dettmers, Ruslan A. Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Q1u25ahSuy.

Xin Ding, Xiaoyu Liu, Zhijun Tu, Yun Zhang, Wei Li, Jie Hu, Hanting Chen, Yehui Tang, Zhiwei Xiong, Baoqun Yin, and Yunhe Wang. CBQ: Cross-block quantization for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Elias Frantar and Dan Alistarh. Sparsegpt: massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

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.

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.

Weiming Hu, Haoyan Zhang, Cong Guo, Yu Feng, Renyang Guan, Zhendong Hua, Zihan Liu, Yue Guan, Minyi Guo, and Jingwen Leng. M-ant: Efficient low-bit group quantization for llms via mathematically adaptive numerical type. In *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 1112–1126. IEEE, 2025.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Qinshuo Liu, Xianglong Liu, Luca Benini, Michele Magno, Shiming Zhang, and XIAOJUAN QI. Slim-LLM: Salience-driven mixed-precision quantization for large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Aojie Jiang, Li Du, and Yuan Du. Groupq: Group-wise quantization with multi-objective optimization for cnn accelerators. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 43 (7):2071–2083, 2024.

Junhan Kim, Gukryeol Lee, Seungwoo Son, Jeewook Kim, and Yongkweon Jeon. Two-stage grid optimization for group-wise quantization of llms. *arXiv preprint arXiv:2602.02126*, 2026.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*, 2023.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. Owq: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13355–13364, 2024.

Deokjae Lee and Hyun Oh Song. Q-palette: Fractional-bit quantizers toward optimal bit allocation for efficient LLM deployment. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Jung Hyun Lee, Jeonghoon Kim, June Yong Yang, Se Jung Kwon, Eunho Yang, Kang Min Yoo, and Dongsoo Lee. Lrq: Optimizing post-training quantization for large language models by learning low-rank weight-scaling matrices. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7708–7743, 2025.

Egor Lifar, Semyon Savkin, Or Ordentlich, and Yury Polyanskiy. Watersic: information-theoretically (near) optimal linear layer quantization. *arXiv preprint arXiv:2603.04956*, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024.

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

Li Lin and Xiaojun Wan. Loaq: Layer-wise output approximation quantization. *arXiv preprint arXiv:2509.06297*, 2025.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. QLLM: Accurate and efficient low-bitwidth quantization for large language models. In *The Twelfth International Conference on Learning Representations*, 2024a.

Ruikang Liu, Yuxuan Sun, Manyi Zhang, Haoli Bai, Xianzhi Yu, Tiezheng Yu, Chun Yuan, and Lu Hou. Quantization hurts reasoning? an empirical study on quantized reasoning models. In *Second Conference on Language Modeling*, 2025.

Yifei Liu, Jicheng Wen, Yang Wang, Shengyu Ye, Li Lyna Zhang, Ting Cao, Cheng Li, and Mao Yang. Vptq: Extreme low-bit vector post-training quantization for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8181–8196, 2024b.

Zheqi Lv, Zhenxuan Fan, Qi Tian, Wenqiao Zhang, and Yueting Zhuang. Enhancing post-training quantization via future activation awareness. *arXiv preprint arXiv:2602.02538*, 2026.

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

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International conference on machine learning*, pp. 7197–7206. PMLR, 2020.

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.

Semyon Savkin, Eitan Porat, Or Ordentlich, and Yury Polyanskiy. Nestquant: nested lattice quantization for matrix products and LLMs. In *Forty-second International Conference on Machine Learning*, 2025.

Claude E Shannon et al. Coding theorems for a discrete source with a fidelity criterion. *IRE Nat. Conv. Rec*, 4(142-163):1, 1959.

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. In *The Twelfth International Conference on Learning Representations*, 2024.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, Wulong Liu, and Jun Yao. FlatQuant: Flatness matters for LLM quantization. In *Forty-second International Conference on Machine Learning*, 2025.

Pingzhi Tang, Ruijie Zhou, Fanxu Meng, Wenjie Pei, and Muhan Zhang. Breaking the blocks: Continuous low-rank decomposed scaling for unified llm quantization and adaptation. *arXiv preprint arXiv:2601.22716*, 2026.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: even better llm quantization with hadamard incoherence and lattice codebooks. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024a.

Albert Tseng, Qingyao Sun, David Hou, and Christopher De Sa. Qtip: quantization with trellises and incoherence processing. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024b. Curran Associates Inc. ISBN 9798331314385.

Mart van Baalen, Andrey Kuzmin, Ivan Koryakovskiy, Markus Nagel, Peter Couperus, Cedric Bastoul, Eric Mahurin, Tijmen Blankevoort, and Paul Whatmough. Gptvq: The blessing of dimensionality for llm quantization. *arXiv preprint arXiv:2402.15319*, 2024.

Shigeng Wang, Chao Li, Yangyuxuan Kang, Jiawei Fan, Zhonghong Ou, and Anbang Yao. Sliderquant: Accurate post-training quantization for LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1648–1665, 2023.

He Xiao, Qingyao Yang, Dirui Xie, Wendong Xu, Zunhai Su, Wenyong Zhou, Haobo Liu, Zhengwu Liu, Ngai Wong, et al. Exploring layer-wise information effectiveness for post-training quantization in small language models. *arXiv preprint arXiv:2508.03332*, 2025.

Yuzhuang Xu, Shiyu Ji, Qingfu Zhu, and Wanxiang Che. Crvq: Channel-relaxed vector quantization for extreme compression of llms. *Transactions of the Association for Computational Linguistics*, 13:1488–1506, 2025a.

Zukang Xu, Xing Hu, Qiang Wu, and Dawei Yang. RSAVQ: Riemannian sensitivity-aware vector quantization for large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b.

Xianglong Yan, ChengZhu Bao, Zhiteng Li, Tianao Zhang, Kaicheng Yang, Haotong Qin, Ruobing Xie, Xingwu Sun, and Yulun Zhang. PT$ˆ2$-LLM: Post-training ternarization for large language models. In *The Fourteenth International Conference on Learning Representations*, 2026.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Jiaming Yang, Chenwei Tang, Caiyang Yu, and Jiancheng Lv. Gwq: Group-wise quantization framework for neural networks. In *Asian Conference on Machine Learning*, pp. 1526–1541. PMLR, 2024.

Sean I Young. Radio: rate-distortion optimization for large language model compression. *arXiv preprint arXiv:2505.03031*, 2025.

Zhihang Yuan, Yuzhang Shang, and Zhen Dong. PB-LLM: Partially binarized large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

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, Shu Yang, Fangmin Chen, Xing Mei, and Lean Fu. Gqsa: Group quantization and sparsity for accelerating large language model inference. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 149–165, 2025.

Aozhong Zhang, Naigang Wang, Yanxia Deng, Xin Li, Zi Yang, and Penghang Yin. Magr: Weight magnitude reduction for enhancing post-training quantization. *Advances in neural information processing systems*, 37:85109–85130, 2024.

Jinhao Zhang, Yunquan Zhang, Daning Chen, Zicheng Yan, et al. Calm: A cka-guided adaptive layer-wise modularization framework for llm quantization. *arXiv preprint arXiv:2512.16282*, 2025.

A Related Work

Despite rapid progress, PTQ for LLMs remains fundamentally fragmented. While existing methods achieve impressive compression ratios, they largely lack a principled framework for preserving task-relevant information under downstream distortion. Most approaches remain magnitude-centric, deriving quantization scales, clipping thresholds, and bit allocations directly from absolute weight values or simple statistical moments of the weight tensors. The classical round-to-nearest (RTN) scheme and its extensions, such as GPTQ Frantar et al. (2023), MagR Zhang et al. (2024), and AWQ Lin et al. (2024) exemplify this paradigm: they minimize local reconstruction error (typically measured by mean-squared error or Hessian-weighted proxies) under the implicit assumption that reducing per-weight deviation is sufficient for maintaining model performance. These techniques achieve high efficiency and strong accuracy at moderate bit widths (4-8 bits). However, this assumption breaks down in aggressive low-bit regimes ($\leq$ 3 bits), where performance collapse is driven primarily by irreversible information loss rather than numerical precision alone.

A parallel line of research adopts an explicitly information-theoretic perspective, framing quantization as a rate-distortion optimization problem that seeks to maximize mutual information or minimize information loss under a downstream distortion constraint. WaterSIC Lifar et al. (2026) introduces a waterfilling-inspired scheme that allocates variable quantization rates across input features according to activation covariance, approaching the theoretical rate-distortion limit for linear layers. EWQ Behtash et al. (2025) and LieQ Xiao et al. (2025) leverage entropy-weighted and layer-wise information-effectiveness metrics to guide mixed-precision allocation, while Radio Young (2025) formulates the entire LLM compression pipeline as a constrained rate-distortion problem. Complementary efforts such as Q-Palette Lee & Song (2025) enable flexible fractional-bit allocation through quantizer palettes and trellis-coded variants, and NestQuant Savkin et al. (2025) exploits nested lattice structures to maximize information capacity. These methods provide a principled view of bit allocation and achieve improved efficiency via adaptive or mixed-precision strategies. Nevertheless, they typically operate at the level of global or layer-wise rate assignment, leaving the actual design of the discrete quantizer underspecified, particularly how codebooks and bin boundaries should be constructed to preserve information under downstream linear transformations

Rather than loading the entire model into memory, the layer-wise quantization paradigm processes one layer (or transformer block) at a time using a small calibration set. As a result, it significantly reduces peak memory consumption and makes optimization feasible even for models with tens or hundreds of billions of parameters. A seminal contribution in this direction is GPTQ Frantar et al. (2023), which introduces a layer-wise greedy reconstruction strategy. Specifically, GPTQ minimizes the output reconstruction error of each linear layer sequentially, while leveraging second-order (Hessian-based) information to compensate for quantization-induced errors. By exploiting the Sherman–Morrison formula for efficient inverse updates, GPTQ achieves strong performance with very low calibration cost—typically requiring only 128–256 samples. Building on this foundation, subsequent work has introduced several key improvements. AdaRound Nagel et al. (2020) replaces standard round-to-nearest operations with learnable, adaptive rounding thresholds. By treating rounding decisions as optimization variables under a reconstruction objective, it substantially improves accuracy, particularly at low bit-widths. LoaQ Lin & Wan (2025) further enhances output fidelity by deriving a closed-form solution for layer-wise output approximation, aligning the quantized outputs more closely with their full-precision counterparts. Meanwhile, CALM Zhang et al. (2025) adopts a CKA (Centered Kernel Alignment)-guided approach to dynamically select the most appropriate PTQ method for each layer, enabling heterogeneous quantization strategies within a single model. Beyond these, a range of methods incorporates sensitivity analysis, activation-aware scaling, or adaptive bit allocation at the layer level. Some recent approaches also combine layer-wise processing with group-wise or channel-wise adaptation, increasing flexibility and improving overall performance.

In parallel, many practical improvements arise from a collection of orthogonal heuristics, each targeting a specific source of quantization error. Group-wise quantization enhances local flexibility by partitioning weights into small column groups, enabling finer-grained scaling and codebook design that better capture local statistics, as seen in GWQ Yang et al. (2024), GroupQ Jiang et al. (2024), GQSA Zeng et al. (2025), M-ANT Hu et al. (2025), TSGQ Kim et al. (2026), SliderQuant Wang et al. (2026), Breaking the Blocks Tang et al. (2026), GlowQ An et al. (2026), and VPTQ Liu et al. (2024b). Complementing this, activation- and sensitivity-aware methods shift the focus to the input side, prioritizing features that contribute most to output distortion by leveraging activation statistics or gradient-based metrics AWQ Lin et al. (2024), FAQ Lv et al. (2026), Astro Chen et al. (2026), RSAVQ Xu et al. (2025b), PT2-LLM Yan et al. (2026), and SliderQuant Wang et al. (2026). In addition, outlier handling techniques explicitly address the heavy-tailed nature of weight distributions by suppressing or selectively restoring high-magnitude entries that disproportionately dominate quantization error via Astro Chen et al. (2026), OS+ Wei et al. (2023), OWQ Lee et al. (2024), and QLLM Liu et al. (2024a). Finally, row-wise and per-channel importance scaling assigns adaptive weights or bit budgets based on intrinsic channel sensitivity, as proposed in HBLLM CHEN et al. (2025), SliM-LLM Huang et al. (2025), AWQ Lin et al. (2024), and CRVQ Xu et al. (2025a). Meanwhile, row-wise and channel-wise scaling strategies introduce another layer of adaptivity by assigning importance

weights or bit budgets based on intrinsic sensitivity across output channels, often improving robustness under aggressive compression.

This separation highlights a deeper limitation in the current PTQ landscape. Existing methods either reason about bit allocation at a global or layer-wise level (information-theoretic approaches) or apply localized error-mitigation heuristics (magnitude-, group-, activation-, and outlier-centric techniques), yet none directly connect the design of the quantizer itself to the joint optimization of information preservation and task-relevant distortion. Consequently, there remains a critical gap for a unified framework that consistently applies a single rate–distortion principle across all stages of quantization, outlier restoration, and pruning.

B DATASETS

We evaluate the quantization performance using both intrinsic language modeling quality and downstream zero-shot reasoning capabilities. For PDGQ, we calibrate using 256 samples randomly drawn from the C4 training set Raffel et al. (2020), each truncated or padded to 2048 tokens. No gradient updates or training data are used beyond this small calibration set.

For perplexity evaluation, we measure perplexity on two widely used benchmarks:

- *WikiText2* Merity et al. (2016): a standard language modeling dataset consisting of over 2 million tokens extracted from high-quality Wikipedia articles.
- *C4* (Colossal Clean Crawled Corpus) Raffel et al. (2020): a large-scale cleaned web corpus containing diverse web text. We use the validation split for evaluation.

Following common practice in LLM quantization works, we use a context length of 2048 tokens for both datasets.

For zero-shot downstream tasks, we employ the *lm-evaluation-harness* framework Biderman et al. (2024) on six standard benchmarks to assess real-world reasoning capability beyond perplexity:

- *PIQA* Bisk et al. (2020): physical commonsense reasoning.
- *HellaSwag* Zellers et al. (2019): commonsense sentence completion.
- *WinoGrande* Sakaguchi et al. (2021): coreference resolution and commonsense reasoning.
- *ARC-Easy* and *ARC-Challenge* Clark et al. (2018): grade-school and challenging science questions.
- *BoolQ* Clark et al. (2019): yes/no question answering.

All evaluations are conducted in a strict zero-shot setting without any few-shot examples.

For open-ended math reasoning (Section 5.4), we evaluate on two benchmarks requiring free-form chain-of-thought generation rather than log-likelihood scoring:

- *GSM8K* Cobbe et al. (2021): grade-school arithmetic word problems, graded on the final numeric answer.
- *MATH-500* Lightman et al. (2024): a 500-problem subset of competition-level mathematics spanning algebra, number theory, geometry, and combinatorics.

Both are graded with `math_verify` (sympy-based symbolic equivalence) on the final `\boxed{}` answer; see Appendix C for the full generation and grading protocol.

C Implementation Details

Hardware and Software Environment. All experiments and quantization processes were conducted using two hardware configurations, depending on model size. Specifically, models with parameter sizes up to 8B were evaluated on a system equipped with an NVIDIA GeForce RTX 3090 GPU and an Intel Xeon Platinum 8272CL CPU running at 2.60GHz, while larger models were evaluated on a high-performance workstation equipped with dual NVIDIA RTX 6000 Ada Generation GPUs (providing a total of 192GB VRAM) and an AMD Ryzen Threadripper processor with 48 physical cores. We observed that using different hardware configurations for the same experiments did not affect the reported results. The software environment was built on CUDA 13.0 and PyTorch 2.x, and we utilized the Hugging Face Transformers library for model loading and management.

Quantization Configuration. For all LLaMA and Qwen3-Base models, we loaded the pre-trained weights in FP16 precision as the baseline. Following the common practice in Post-Training Quantization (PTQ), we used a calibration set consisting of 256 samples randomly selected from the C4 dataset, each with a sequence length of 2048 tokens. Our PDGQ method was applied with a group size of 128. For the Lloyd-based refinement process, we set the number of iterations to $J = 20$, and entropy weight to $\gamma = 0.01$. The PDOR threshold was fixed at $\alpha = 0.1$ across all experiments to balance the trade-off between outlier preservation and compression efficiency. The Qwen3-Base experiments use exactly the same configuration as the LLaMA experiments.

Baselines. Baseline results are quoted from the original publications where they report the same protocol (WikiText2 and C4 perplexity at context length 2048, and the six zero-shot benchmarks); entries that are not reported there were obtained by re-running the baseline under the same protocol.

Math Reasoning (WxAxKVx) Configuration. For the GSM8K/MATH-500 results (Section 5.4), the weight codebook uses the same calibration protocol as above (256 C4 samples, 2048-token chunks, group size 128, $J = 20$, $\alpha = 0.1$), with entropy weight $\gamma = 0$. Activations and the KV-cache are quantized independently of the weight codebook via round-to-nearest: activations and the K-cache use per-channel scaling with a running max-abs tracked across the whole generation (a fresh per-call or per-token scale is dominated by a small number of outlier channels, crushing every other channel to zero), while the V-cache uses per-token round-to-nearest (KIVI-style). Because these running statistics must not leak across unrelated examples, generation is performed one example at a time rather than batched. Decoding uses temperature 0.6, top-p 0.95, and up to 32768 generated tokens per example, with the prompt *"{question}\n\nPlease reason step by step, and put your final answer within \boxed{}."* Correctness is graded with `math_verify` (sympy-based symbolic equivalence) applied to the *last* `\boxed{}` expression in the generation, so that self-correcting reasoning traces that revise an earlier boxed answer are graded on the final one rather than merged across all occurrences.

D Technical appendices and supplementary material

D.1 Derivation of Expected Output Distortion

We derive the approximation of the expected output distortion induced by quantization. Let $R = W - \hat{W}$ denote the quantization residual matrix. The output residual is given by:

$$y - \hat{y} = (W - \hat{W})\mathbf{x} = R\mathbf{x} \tag{11}$$

The expected squared error is:

$$\mathbb{E}\left[|E\mathbf{x}|^2\right] = \mathbb{E}\left[\mathbf{x}^\top E^\top E\mathbf{x}\right]. \tag{12}$$

Using the identity $\mathbb{E}[\mathbf{x}^\top A\mathbf{x}] = \mathrm{Tr}(A\Sigma_x)$, where $\Sigma_x = \mathbb{E}[\mathbf{x}\mathbf{x}^\top]$ is the (uncentered) second-moment matrix of the input activations, we obtain:

$$\mathbb{E}\left[|E\mathbf{x}|^2\right] = \mathrm{Tr}(E^\top E\Sigma_x). \tag{13}$$

By cyclic permutation of the trace, this can be rewritten as:

$$\mathrm{Tr}(E^\top E\,\Sigma_x) = \mathrm{Tr}(EE^\top\Sigma_x), \tag{14}$$

which matches the form used in the main text.

To make the expression more interpretable, we assume that input features are approximately uncorrelated, so that the covariance matrix is diagonal:

$$\Sigma_x = \mathrm{diag}(\sigma_x^2(0), \ldots, \sigma_x^2(I-1)). \tag{15}$$

In this derivation $\sigma_x^2(i) = [\Sigma_x]_{ii}$, while PDGQ estimates it by the centered per-column variance (Appendix D.2).

Under this assumption, the trace can be expanded as:

$$\mathrm{Tr}(E^\top E\,\Sigma_x) = \sum_{i=1}^{I} \sigma_x^2(i)\,(E^\top E)_{i,i} \tag{16}$$

$$= \sum_{i=1}^{I} \sigma_x^2(i) \sum_{o=1}^{O} e_{oi}^2 \tag{17}$$

where $(E^\top E)_{i,i}$ corresponds to the sum of squared residual in the i-th input dimension.

Rearranging the summation yields:

$$\mathbb{E}\left[|E\mathbf{x}|^2\right] \approx \sum_{o=1}^{O}\sum_{i=1}^{I} e_{oi}^2\,\sigma_x^2(i), \tag{18}$$

where e_{oi} denotes the (o,i)-th element of the residual matrix R.

D.2 Empirical Check of the Diagonal Approximation

The per-weight objective replaces the exact distortion $\mathrm{Tr}(E\,\Sigma_x E^\top)$ by a diagonal estimate, so its error has two sources, the omitted cross terms $\sum_{i\neq j} e_{oi}e_{oj}[\Sigma_x]_{ij}$ and the centering of each column. We measure both on Qwen2.5-0.5B-Instruct and DeepSeek-R1-Distill-Qwen-1.5B. For five layers spread evenly over depth (first, quarter, middle, three-quarter, last) and all seven linear layers in each, we accumulate the full second-moment matrix Σ_x from 128 calibration sequences of 2048 tokens, quantize the layer with the production PDGQ code ($\alpha = 0.1$, group size 128, $J = 20$, $\gamma = 0$), and take E to be the final residual, that is, zero at restored weights and w_{oi} at pruned ones. Table 4 compares the exact $\mathrm{Tr}(E\,\Sigma_x E^\top)$ with two diagonal estimates, the diagonal of Σ_x itself, $\mathbb{E}[x_i^2]$, and the centered variance $\mathbb{E}[x_i^2] - \mathbb{E}[x_i]^2$ used by PDGQ.

Model	Bits	$\mathbb{E}[x_i^2]$ / exact	Variance / exact	ρ (diag.)	ρ (unweighted)
Qwen2.5-0.5B-Instruct	2	0.99 (0.70–1.01)	0.92 (0.54–0.98)	0.90	0.90
	3	1.00 (0.85–1.02)	0.92 (0.60–0.98)	0.90	0.90
	4	1.00 (0.96–1.04)	0.93 (0.58–0.98)	0.91	0.91
DeepSeek-R1-Distill-Qwen-1.5B	2	0.99 (0.70–1.00)	0.90 (0.44–0.97)	0.85	0.83
	3	1.00 (0.88–1.02)	0.90 (0.50–0.98)	0.84	0.83
	4	1.00 (0.95–1.03)	0.90 (0.50–0.98)	0.83	0.83

Table 4: Diagonal estimate of the projected distortion divided by the exact $\mathrm{Tr}(E\,\Sigma_x E^\top)$, reported as median (min–max) over 35 linear layers per model and bit-width. ρ is the median Spearman correlation across output rows between the exact per-row distortion and the diagonal estimate (variance), or the unweighted $\|e_o\|^2$ for reference.

With $\mathbb{E}[x_i^2]$ on the diagonal, the estimate recovers the exact distortion to within 1% at the median for every bit-width, and the omitted cross terms stay within about 5% at $b = 4$ but grow at lower bit-widths. The worst cases are concentrated in the first and last layers at 2 bits, for example 30% underestimation for the last-layer value projection of Qwen2.5-0.5B and the last-layer output projection of DeepSeek-R1-Distill-Qwen-1.5B. The centered variance is a systematically smaller quantity, because LLM activations carry a non-zero mean in a few channels, so it sits at about $0.9\times$ the exact value at the median and as low as $0.44\times$ in the worst layer. Whether this matters for quality is tested directly in Table 5, where every decoder linear layer is quantized with each choice. The variance gives equal or lower WikiText2 perplexity in all six settings, most clearly for DeepSeek-R1-Distill-Qwen-1.5B at 2 bits (44.8 against 49.5), so PDGQ keeps the centered variance. The row-level ranking of distortion agrees well with the exact one ($\rho \approx 0.83$–0.91), although unweighted error already ranks rows nearly as well, so the benefit of the activation-weighted objective does not come from row ranking alone.

Model (FP16 perplexity)	Bits	Variance (PDGQ)	$\mathbb{E}[x_i^2]$
Qwen2.5-0.5B-Instruct (14.24)	2	21.61	21.67
	3	15.45	15.51
	4	14.44	14.47
DeepSeek-R1-Distill-Qwen-1.5B (40.45)	2	44.83	49.53
	3	43.82	44.78
	4	40.50	40.63

Table 5: WikiText2 perplexity when the activation energy $\sigma_x^2(i)$ is the centered variance (used by PDGQ) or the uncentered second moment $\mathbb{E}[x_i^2]$, with every other setting identical (all decoder linear layers, $\alpha = 0.1$, $J = 20$, $\gamma = 0$). Single run per setting.

Block- and full-covariance objectives. To test whether a more exact objective would change the result, we replace the diagonal Lloyd stage by a refinement against the exact projected distortion $\mathrm{Tr}(E\,\Sigma_x E^\top)$. Starting from the PDGQ solution, we run three sweeps of block coordinate descent over column groups, each sweep alternating an assignment pass over the columns of the group with a closed-form solve for every row's K codewords, so that each step minimizes its sub-problem exactly. In the *block* variant Σ_x is restricted to its 128×128 diagonal blocks, one per quantization group, and in the *full* variant the whole matrix is used, which is the layer-wise Hessian objective of GPTQ up to scaling. PDOR and DCP then run unchanged on the refined reconstruction. Table 6 reports WikiText2 perplexity for the codebook stage alone, where the bit-width is exactly b, and for the complete pipeline.

Model (FP16)	b	Codebook only (b bits) Diag.	Block	Full	Full pipeline Diag.	Block	Full
Qwen2.5-0.5B (14.24)	2	2239	519	727	21.60	20.64	20.04
	3	35.56	26.62	27.58	15.45	15.57	15.42
	4	19.03	17.69	17.22	14.44	14.49	14.53
DeepSeek-R1-Distill-1.5B (40.45)	2	1704	383	209	44.86	46.07	44.80
	3	83.21	78.17	63.75	43.82	45.56	44.58

Table 6: WikiText2 perplexity when the codebook stage minimizes the diagonal (PDGQ), block-diagonal, or full-covariance projected distortion, with $\alpha = 0.1$, $J = 20$, $\gamma = 0$ and three refinement sweeps for Block and Full. Single run per setting, so the diagonal column can differ from Table 5 in the second decimal; the DeepSeek 4-bit setting was not run.

At the codebook stage the exact objective helps substantially, for example from 1704 to 383 (block) and 209 (full) for DeepSeek-R1-Distill-Qwen-1.5B at 2 bits and from 35.6 to 26.6 for Qwen2.5-0.5B at 3 bits, although the full covariance is not consistently better than the block one. In the complete pipeline the gain disappears. Perplexity differs from the diagonal PDGQ by less than two points in either direction with no consistent winner, even though the exact objective leaves more weights above the PDOR threshold (18–26% restored against 14–15% for the diagonal objective). PDOR and DCP therefore absorb most of what a more exact codebook objective would provide, which is consistent with the diagonal estimate being sufficient at the operating point of PDGQ. This is a limited test, since it uses a local three-sweep solver on models of at most 1.5B parameters.

D.3 Rate–Distortion Analysis for Gaussian Weight Quantization

We model a layer's neural network weights as a random vector

$$\mathbf{W} \in \mathbb{R}^d, \quad \mathbf{W} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \tag{19}$$

where $\mathbf{\Sigma} \in \mathbb{R}^{d\times d}$ is a symmetric positive-definite covariance matrix. Let $\hat{\mathbf{W}}$ denote the quantized reconstruction of $\mathbf{W}$, obtained via a conditional distribution $p_{\hat{\mathbf{W}}|\mathbf{W}}(\hat{\mathbf{w}}|\mathbf{w})$. Throughout, uppercase letters denote random vectors and lowercase letters their realizations.

D.3.1 Setup

Distortion is measured by mean squared error:

$$d(\mathbf{w}, \hat{\mathbf{w}}) = \|\mathbf{w} - \hat{\mathbf{w}}\|_2^2, \qquad \mathbb{E}\big[\|\mathbf{W} - \hat{\mathbf{W}}\|_2^2\big] \le D. \tag{20}$$

The rate-distortion function gives the minimum mutual information (equivalently, the minimum description rate in bits) needed to encode $\mathbf{W}$ up to distortion D:

$$R(D) \;=\; \inf_{\substack{p_{\hat{\mathbf{W}}|\mathbf{W}} \\ \mathbb{E}[\|\mathbf{W}-\hat{\mathbf{W}}\|_2^2] \le D}} I(\mathbf{W}; \hat{\mathbf{W}}). \tag{21}$$

D.3.2 Differential Entropy of Weight

The differential entropy of $\mathbf{W}$ with density $p_{\mathbf{W}}$ is

$$h(\mathbf{W}) = -\int p_{\mathbf{W}}(\mathbf{w}) \log_2 p_{\mathbf{W}}(\mathbf{w})\, d\mathbf{w}. \tag{22}$$

For $\mathbf{W} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ with density

$$p_{\mathbf{W}}(\mathbf{w}) = \frac{1}{(2\pi)^{d/2} \det(\mathbf{\Sigma})^{1/2}} \exp\Big(-\tfrac{1}{2}\mathbf{w}^\top \mathbf{\Sigma}^{-1}\mathbf{w}\Big), \tag{23}$$

a standard calculation gives

$$h(\mathbf{W}) = \tfrac{1}{2}\log_2\Big[(2\pi e)^d \det(\mathbf{\Sigma})\Big]. \tag{24}$$

D.3.3 Diagonalization and Decomposition

Diagonalize $\mathbf{\Sigma}$ via its eigendecomposition:

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top, \qquad \mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_d), \quad \lambda_i > 0, \tag{25}$$

where $\mathbf{U} \in \mathbb{R}^{d\times d}$ is orthogonal ($\mathbf{U}^\top\mathbf{U} = \mathbf{I}$). Define the rotated variables

$$\mathbf{V} = \mathbf{U}^\top \mathbf{W} \in \mathbb{R}^d, \qquad \hat{\mathbf{V}} = \mathbf{U}^\top \hat{\mathbf{W}} \in \mathbb{R}^d. \tag{26}$$

Since $\mathbf{U}$ is an orthogonal (volume-preserving) transformation, it is a bijection that leaves joint densities invariant up to a Jacobian of one. Consequently, it preserves both mutual information and squared-error distortion:

$$I(\mathbf{W}; \hat{\mathbf{W}}) = I(\mathbf{V}; \hat{\mathbf{V}}), \qquad \|\mathbf{W} - \hat{\mathbf{W}}\|_2^2 = \|\mathbf{V} - \hat{\mathbf{V}}\|_2^2. \tag{27}$$

The covariance of $\mathbf{V}$ is $\mathbf{U}^\top \mathbf{\Sigma}\, \mathbf{U} = \mathbf{\Lambda}$, so the components of $\mathbf{V}$ are *mutually independent* Gaussians:

$$V_i \sim \mathcal{N}(0, \lambda_i), \quad i = 1, \ldots, d. \tag{28}$$

D.3.4 Scalar Rate-Distortion Function

We use the classical result for a scalar Gaussian. Let $V \sim \mathcal{N}(0,\, \lambda)$ and let $\hat{V}$ be its reconstruction under a squared-error constraint. The rate-distortion function is

$$R(D_0) = \max\left\{0,\ \tfrac{1}{2}\log_2\left(\frac{\lambda}{D_0}\right)\right\}. \tag{29}$$

For $D_0 \geq \lambda$, zero bits suffice: the trivial encoder $\hat{V} \equiv 0$ already achieves $\mathbb{E}[(V - \hat{V})^2] = \lambda \leq D_0$. For $D_0 < \lambda$, the minimum rate is $\frac{1}{2}\log_2(\lambda/D_0) > 0$.

Applying equation 29 to each independent component $V_i \sim \mathcal{N}(0, \lambda_i)$ with per-component distortion budget D_i:

$$R_i(D_i) = \max\left\{0,\ \tfrac{1}{2}\log_2\left(\frac{\lambda_i}{D_i}\right)\right\}, \qquad D_i \geq 0. \tag{30}$$

D.3.5 Decomposition into Independent Scalar Problems

Because $V_1, \ldots, V_d$ are mutually independent, the joint rate-distortion problem decomposes additively:

$$R(D) = \min_{\substack{D_1,\ldots,D_d \geq 0 \\ \sum_{i=1}^d D_i \leq D}} \sum_{i=1}^d R_i(D_i) = \min_{\substack{D_i \geq 0 \\ \sum_i D_i \leq D}} \sum_{i=1}^d \max\left\{0,\ \tfrac{1}{2}\log_2\left(\frac{\lambda_i}{D_i}\right)\right\}. \tag{31}$$

D.3.6 WATER-FILLING SOLUTION

We solve equation 31 via the KKT conditions. Let $\mathcal{A} = \{i : D_i < \lambda_i\}$ be the set of *active* components (those that require a positive rate). On $\mathcal{A}$, the objective is smooth and strictly convex in each D_i. Forming the Lagrangian for the inequality constraint $\sum_i D_i \le D$ with multiplier $\mu \ge 0$:

$$\mathcal{L}(\{D_i\}, \mu) = \sum_{i \in \mathcal{A}} \frac{1}{2} \log_2\left(\frac{\lambda_i}{D_i}\right) + \mu\left(\sum_{i=1}^{d} D_i - D\right). \tag{32}$$

Setting $\partial \mathcal{L}/\partial D_i = 0$ for each $i \in \mathcal{A}$:

$$-\frac{1}{2D_i \ln 2} + \mu = 0 \quad \Longrightarrow \quad D_i = \frac{1}{2\mu \ln 2}. \tag{33}$$

This is constant across all active components. Defining $\theta := \frac{1}{2\mu \ln 2} > 0$, the optimal per-component distortion is

$$D_i^* = \min(\lambda_i,\, \theta), \tag{34}$$

where the $\min$ encodes the two cases: component i is active ($D_i^* = \theta < \lambda_i$) or discarded ($D_i^* = \lambda_i \le \theta$, contributing zero rate). The water level $\theta > 0$ is determined by the binding total distortion constraint:

$$\sum_{i=1}^{d} \min(\lambda_i,\, \theta) = D. \tag{35}$$

Interpretation. The water level θ partitions the principal components into two groups. Components with $\lambda_i > \theta$ are *encoded*: each receives distortion θ and contributes $\frac{1}{2}\log_2(\lambda_i/\theta) > 0$ bits, with higher-variance directions automatically receive more bits. Components with $\lambda_i \le \theta$ are *discarded*: they receive no bits ($R_i = 0$) and their full variance λ_i is absorbed into the distortion budget.

This is the information-theoretic justification for *PCA-based quantization*: rotating into the eigenbasis of $\mathbf{\Sigma}$, truncating low-variance directions, and quantizing the remainder is asymptotically optimal. In the limit where all components are active ($\theta < \min_i \lambda_i$), no truncation occurs and the full geometric-mean rate-distortion tradeoff equation 42 applies.

D.3.7 RATE-DISTORTION FUNCTION

Substituting $D_i^* = \min(\lambda_i, \theta)$ into equation 31 and noting that $\frac{1}{2}\log_2(\lambda_i/\theta) = 0$ whenever $\lambda_i \le \theta$:

$$\boxed{R(D) = \frac{1}{2}\sum_{i=1}^{d} \max\left\{0,\ \log_2\left(\frac{\lambda_i}{\theta}\right)\right\},} \tag{36}$$

where $\theta \ge 0$ satisfies $\sum_{i=1}^{d} \min(\lambda_i, \theta) = D$.

D.3.8 OPTIMAL TEST CHANNEL

The achieved conditional distribution is Gaussian. In the eigenbasis:

$$V_i = \hat{V}_i + Z_i, \qquad Z_i \sim \mathcal{N}(0, D_i^*), \quad Z_i \perp \hat{V}_i, \quad i = 1, \ldots, d. \tag{37}$$

Rotating back to the original basis via $\mathbf{W} = \mathbf{U}\mathbf{V}$:

$$\mathbf{W} = \hat{\mathbf{W}} + \mathbf{Z}, \qquad \mathbf{Z} \sim \mathcal{N}(\mathbf{0},\, \mathbf{K}_Z), \qquad \mathbf{K}_Z = \mathbf{U}\ \mathrm{diag}(D_1^*, \ldots, D_d^*)\, \mathbf{U}^\top. \tag{38}$$

This achieves $\mathbb{E}[\|\mathbf{W} - \hat{\mathbf{W}}\|_2^2] = \mathrm{tr}(\mathbf{K}_Z) = \sum_i D_i^* = D$, and $I(\mathbf{W}; \hat{\mathbf{W}})$ matches the bound in equation 36.

D.3.9 HIGH-RATE APPROXIMATION

When b bits per dimension are used, the total rate is $R = d \cdot b$ bits. In the high-rate regime ($b \to \infty$, equivalently $\theta \to 0$), all $\lambda_i > \theta$ so every component is active. The water-filling sum becomes a full sum:

$$R = d \cdot b = \frac{1}{2} \sum_{i=1}^{d} \log_2 \left(\frac{\lambda_i}{\theta} \right) = \frac{1}{2} \log_2 \left(\frac{\det(\mathbf{\Sigma})}{\theta^d} \right). \tag{39}$$

Since all d components are active, $D_i^* = \theta$ for all i, so $D = \sum_i D_i^* = d\theta$, giving $\theta = D/d$. It is natural to work directly with the *per-weight distortion* $\bar{D} := D/d$, so that $\theta = \bar{D}$ exactly. Substituting $\theta = \bar{D}$:

$$d \cdot b = \frac{1}{2} \log_2 \left(\frac{\det(\mathbf{\Sigma})}{\bar{D}^{\,d}} \right) = \frac{d}{2} \log_2 \left(\frac{\det(\mathbf{\Sigma})^{1/d}}{\bar{D}} \right). \tag{40}$$

This is *exact* in the high-rate regime (no terms are dropped). Dividing both sides by d gives the per-weight rate–distortion relationship:

$$b = \frac{1}{2} \log_2 \left(\frac{\det(\mathbf{\Sigma})^{1/d}}{\bar{D}} \right), \tag{41}$$

and inverting:

$$D^*(b) = \det(\mathbf{\Sigma})^{1/d} \cdot 2^{-2b} = \left(\prod_{i=1}^{d} \lambda_i \right)^{1/d} \cdot 2^{-2b}, \tag{42}$$

where $\det(\mathbf{\Sigma})^{1/d} = \left(\prod_i \lambda_i \right)^{1/d}$ is the *geometric mean* of the eigenvalues. This captures the effective per-weight scale of the distribution: anisotropic weights (large spread in eigenvalues) are harder to quantize than isotropic ones of the same total variance.

Gaussian weights are the hardest to compress Among all distributions with a fixed covariance $\mathbf{\Sigma}$, the Gaussian maximizes differential entropy and thus maximizes $R(D)$. If the true weight distribution has heavier tails (e.g. Laplace or Student-t, as observed empirically in deep networks), the actual rate-distortion function lies *below* the Gaussian bound. Consequently, equation 42 serves as a *conservative* estimate: in practice, weights may be compressible beyond what this formula predicts.

D.3.10 COROLLARY: PER-WEIGHT DISTORTION FOR A GAUSSIAN WEIGHT MATRIX

Let $\mathbf{W} \in \mathbb{R}^{O \times I}$ be a weight matrix whose entries are i.i.d. $\mathbf{W}_{oi} \sim \mathcal{N}(0, \sigma_w^2)$. Flatten $\mathbf{W}$ into a random vector $\mathbf{w} \in \mathbb{R}^d$ with $d = O \times I$. Since the entries are independent and identically distributed, the covariance is $\mathbf{\Sigma} = \sigma_w^2 \mathbf{I}_d$, so all eigenvalues are equal: $\lambda_i = \sigma_w^2$ for all $i = 1, \ldots, d$.

Each entry is quantized independently using b bits, so the total rate is $R = d \cdot b$ bits. With $\lambda_i = \sigma_w^2$ for all i, the water-filling constraint $\sum_{i=1}^{d} \min(\lambda_i, \theta) = D$ reduces to $d\theta = D$, i.e., $\theta = D/d = \bar{D}$, and every component is active for any $b > 0$. The rate–distortion function equation 36 becomes

$$d \cdot b = \frac{d}{2} \log_2 \left(\frac{\sigma_w^2}{\bar{D}} \right). \tag{43}$$

Solving for $\bar{D}$ directly:

$$\boxed{D^*(b) = \sigma_w^2 \cdot 2^{-2b}.} \tag{44}$$

This result is *exact* (no approximation): when $\mathbf{\Sigma} = \sigma_w^2 \mathbf{I}_d$, the water level $\theta = \bar{D}$ holds exactly and $\det(\mathbf{\Sigma})^{1/d} = \sigma_w^2$, so equation 42 is tight. The per-weight MSE decays geometrically as 2^{-2b} with bit-width and scales linearly with σ_w^2, establishing the *intrinsic noise floor* of quantization at b bits per weight.

The rate-distortion bound $D^*(b) = \sigma_w^2 \cdot 2^{-2b}$ is a fundamental lower bound on per-weight MSE achievable by *any* scheme, including vector quantizers with unlimited block length. Thus $D^*(b)$ should be interpreted as the theoretical minimum, not a directly achievable target for fixed-width uniform quantizers.

D.3.11 JUSTIFICATION OF THE GAUSSIAN RESIDUAL MODEL USED BY PDOR

Section 4.2's null hypothesis models a weight's quantization residual as $e_{oi} \sim \mathcal{N}(0, D^*(b, g))$. This is not an independent modeling choice: it is the scalar specialization, to the i.i.d.-Gaussian-per-row-group setting of the Corollary above, of the optimal test channel derived in Section D.3.8. Under $\mathbf{\Sigma} = \sigma_w^2(g)\mathbf{I}$ (all weights in row-group g equally scaled), the eigenbasis is the standard basis, so the rotation $\mathbf{U}$ in equation 31–equation 36 is the identity, every eigenvalue equals $\sigma_w^2(g)$, and the water level is the same for every component: $D_i^* = \theta = D^*(b, g)$. The test-channel result $Z_i \sim \mathcal{N}(0, D_i^*)$, $Z_i \perp \hat{V}_i$, therefore applies directly to *each individual weight* w_{oi} in the group rather than only to a rotated aggregate, giving

$$e_{oi} = w_{oi} - \hat{w}_{oi} \;\sim\; \mathcal{N}\big(0,\, D^*(b, g)\big), \qquad \text{independent across } (o, i), \tag{45}$$

which is exactly the null hypothesis used to build the $\chi^2(1)$ test statistic in Section 4.2. In this sense the Gaussian residual model has a real derivation, not just an assumption pulled from convention.

Where the derivation stops applying. Two gaps separate this result from the residual actually produced by PDOR in practice, both already flagged qualitatively in Section M. First, the test channel above is the residual of an *optimal* rate-distortion coupling – realizing it in general requires the asymptotic, unbounded block-length random-coding argument behind Shannon's achievability proof. The codebook actually used (Section 4.1) is a practical, finite-dimensional (group size 128), deterministic Lloyd quantizer, and nothing guarantees its residual reproduces the optimal test channel exactly. Second, the source is only *approximately* Gaussian per row-group: real LLM weights are heavier-tailed, as noted after equation 42. The Gaussian residual model is therefore best read as the theoretically principled anchor the test is built on, not an exact description of the deployed quantizer – it is why α has a clean statistical interpretation at all, rather than why the resulting threshold is exact.

Empirical check. We can measure directly how large this gap is. Using the real per-weight test statistic T_{oi} extracted from the layer in Figure 1, we compare its empirical distribution to the $\chi^2(1)$ it would follow under an exact Gaussian residual:

	median T_{oi} (empirical / theoretical)	$P(T_{oi} \le \chi^2_{1-\alpha}(1))$ empirical	theoretical $(1-\alpha)$
$b = 2$	0.786 / 0.455	90.7%	95.0%
$b = 3$	0.644 / 0.455	88.8%	95.0%

The empirical median exceeds the theoretical $\chi^2(1)$ median at both bit-widths, and the observed fraction of weights below the nominal threshold falls short of $1 - \alpha = 95\%$ – i.e. the practical Lloyd quantizer's residual is somewhat more dispersed than an exact Gaussian null would predict, so on this layer PDOR's true flagging rate runs above its nominal α (matching the 9–11% outlier rates reported alongside Figure 1 against a nominal $\alpha = 5\%$). This is consistent with, and quantifies, the two gaps above: the assumption is a principled anchor rather than an exact fit, and in practice it makes PDOR mildly *more* conservative (flags more weights) than α alone would suggest, rather than less.

D.4 Deterministic Quantization Implies Zero Conditional Entropy

Let $\hat{W} = Q(W)$ be a deterministic quantization mapping from a continuous random variable W to a discrete random variable $\hat{W}$. By definition, the conditional entropy of $\hat{W}$ given W is:

$$H(\hat{W}|W) = \mathbb{E}\left[-\sum_{\hat{w}} P(\hat{w}|W)\log P(\hat{w}|W)\right] \tag{46}$$

For a deterministic mapping, the conditional distribution $P(\hat{W}|W)$ is degenerate:

$$P(\hat{w} \mid W) = \begin{cases} 1, & \text{if } \hat{w} = Q(W), \\ 0, & \text{otherwise.} \end{cases} \tag{47}$$

Substituting into the definition of conditional entropy, we obtain

$$H(\hat{W}|W) = \mathbb{E}_W[-\log 1] = 0 \tag{48}$$

D.5 Entropy Maximization under Probability Constraints

Proposition. Let $\{p_k\}_{k=1}^K$ be a discrete distribution with $\sum_{k=1}^K p_k = 1$. The entropy

$$H(\hat{W}) = -\sum_{k=1}^{K} p_k \log_2 p_k \tag{49}$$

is maximized when $p_k = \frac{1}{K}$ for all k, in which case $H(\hat{W}) = \log_2 K$.

Proof. Consider the constrained optimization problem:

$$\max_{\{p_k\}} -\sum_{k=1}^{K} p_k \log_2 p_k \quad \text{s.t.} \quad \sum_{k=1}^{K} p_k = 1 \tag{50}$$

We form the Lagrangian:

$$\mathcal{L} = -\sum_{k=1}^{K} p_k \log_2 p_k + \lambda\left(\sum_{k=1}^{K} p_k - 1\right) \tag{51}$$

Taking the derivative with respect to p_k and setting to zero:

$$\frac{\partial \mathcal{L}}{\partial p_k} = -\left(\log_2 p_k + \frac{1}{\ln 2}\right) + \lambda = 0, \tag{52}$$

which implies that $\log p_k$ is constant across all k, and hence $p_k = c$. Enforcing the normalization constraint yields $c = \frac{1}{K}$, so $p_k = \frac{1}{K}$ for all k. Substituting, we obtain:

$$H(\hat{W}) = -\sum_{k=1}^{K} \frac{1}{K}\log_2 \frac{1}{K} = \log_2 K \tag{53}$$

Finally, since entropy is a strictly concave function over the probability simplex, this stationary point corresponds to the global maximum.

E Implementation

The quantization process is sketched as in Algorithm 4.

Algorithm 1 Projected-Distortion for Codebook Optimization

function PDCODEOPT($W, \sigma_x^2, b, J, \gamma$)
  $K \leftarrow 2^b$
  Init: K equiprobable bins via quantiles.
  Initialize $\{c_k\}$, assignments $\{a_i\}$, and $\{p_k\}$.
  for $\ell = 1, \ldots, J$ **do**
    $a_i \leftarrow \arg\min_k \left[(W_i - c_k)^2 \sigma_x^2(i) + \gamma \log p_k\right]$
    $p_k \leftarrow \frac{1}{|\mathcal{W}|} \sum_i \mathbf{1}[a_i = k]$
    $c_k \leftarrow \frac{\sum_{i:a_i=k} \sigma_x^2(i)\, W_i}{\sum_{i:a_i=k} \sigma_x^2(i)}$
  end for
  return $\hat{W}_i = c_{a_i}$
end function

Algorithm 3 Distortion-Consistent Pruning

function DCP($W, \hat{W}, \mathcal{R}$)
  $e \leftarrow W - \hat{W}$
  for each $(i, j) \notin \mathcal{R}$ **do**
    if $|W_{ij}| \leq |e_{ij}|$ **then**
      $\hat{W}_{ij} \leftarrow 0$
    end if
  end for
  return $\hat{W}$
end function

Algorithm 2 Per-weight Projected-Distortion for Outlier Restoration

function PDOR($W, \hat{W}, \sigma_x^2, b, \alpha$)
  $R \leftarrow W - \hat{W}$
  $D^*_{i,g} \leftarrow \mathrm{Var}(W_{i,g}) \cdot 2^{-2b}$ for each row-group (i, g)
  $\bar{\sigma}_x^2(g) \leftarrow \mathrm{mean}_{j \in g}\, \sigma_x^2(j)$
  $T_{ij} \leftarrow \dfrac{r_{ij}^2 \cdot \sigma_x^2(j)}{D^*_{i,g(j)} \cdot \bar{\sigma}_x^2(g(j))}$
  $\mathcal{R} \leftarrow \{(i, j) : T_{ij} > \chi^2_{1-\alpha}(1)\}$
  $\hat{W}_{ij} \leftarrow W_{ij}$ **for all** $(i, j) \in \mathcal{R}$
  return $\hat{W}$, $\mathcal{R}$
end function

Algorithm 4 PDGQ Main Pipeline

Require: $W \in \mathbb{R}^{O \times I}$, σ_x^2, b, J, α, γ
for $o = 1, \ldots, O$ **do**
  $\mu_o \leftarrow \mathrm{mean}(|W_{o,:}|)$
  $W_o \leftarrow W_o \oslash \mu_o$
  for each group g **do**
    $\hat{W}_{o,g} \leftarrow$ PDCODEOPT($W_{o,g}$, $\sigma_x^2(g)$, b, J, γ)
  end for
  $\hat{W}_o \leftarrow \hat{W}_o \odot \mu_o$
end for
$(\hat{W}, \mathcal{R}) \leftarrow$ PDOR(W, $\hat{W}$, σ_x^2, b, α)
$\hat{W} \leftarrow$ DCP(W, $\hat{W}$, $\mathcal{R}$)
return $\hat{W}$

F AVERAGE BIT-WIDTH COMPUTATION

Let b denote the base quantization bit-width, p_{out} the fraction of weights restored to full precision by PDOR, and p_{sparse} the fraction pruned by DCP. A decoder must know *which* positions hold restored FP16 scalars or pruned zeros before it can reconstruct $\hat{W}$, so the effective bit-width per weight has to account for both the values stored and the cost of locating them. Within a row-group of size 128 (Section 4.1), a single position can be addressed with a $\lceil \log_2 128 \rceil = 7$-bit sparse index. Weights retained at b-bit precision (a fraction $1 - p_{\text{out}} - p_{\text{sparse}}$) cost b bits each. Each restored weight costs 16 bits for its FP16 value plus 7 bits for its index, and pruned weights store no value at all. Restored fractions are always small in practice ($p_{\text{out}} \leq 1.81\%$ across Table 10), so sparse indexing is always the cheaper way to locate them. Pruned positions are more numerous, so we charge the *cheaper* of the same 7-bit sparse index or a flat 1-bit-per-weight bitmask covering the whole group, which costs 1 bit per weight regardless of p_{sparse} and therefore

#bits	Method	1-7B	1-13B	1-30B	1-65B	2-7B	2-13B	2-70B	3-8B	3-70B
16	-	5.68	5.09	4.10	3.53	5.47	4.88	3.31	5.75	2.90
4-bit	RTN	6.29	5.53	4.54	3.92	6.02	5.24	3.56	6.91	3.32
	AWQ Lin et al. (2024)	5.81	5.16	4.19	3.61	5.58	4.94	3.39	6.38	3.10
	GPTQ Frantar et al. (2023)	5.85	5.20	4.23	3.65	5.62	4.98	3.42	6.45	3.15
	CBQ Ding et al. (2025)	5.72	5.14	4.14	3.59	**5.52**	4.94	3.42	6.88	3.10
	SliderQuant Wang et al. (2026)	5.71	5.11	4.14	3.58	5.61	5.00	3.41	6.79	3.06
	Radio Young (2025)	5.73	5.13	4.15	3.60	5.57	4.97	3.40	6.91	3.12
	NestQuant Savkin et al. (2025)	**5.70**	**5.10**	4.15	3.60	5.53	4.93	3.38	6.31	3.14
	LRQ Lee et al. (2025)	5.74	5.12	4.16	3.61	5.75	5.02	3.47	6.94	3.20
	PDGQ	5.72	5.12	**4.13**	**3.56**	**5.52**	**4.90**	**3.34**	**6.23**	**2.94**
3-bit	RTN	7.01	5.88	4.87	4.24	6.66	5.51	3.97	27.91	11.84
	AWQ Lin et al. (2024)	6.46	5.51	4.63	3.99	6.24	5.32	-	8.22	4.81
	GPTQ Frantar et al. (2023)	6.55	5.62	4.80	4.17	6.29	5.42	3.85	8.19	5.22
	SliM-LLM Huang et al. (2025)	6.40	5.48	4.61	3.99	6.24	5.26	3.67	7.16	4.08
	CBQ Ding et al. (2025)	6.42	5.50	4.63	4.01	5.89	5.23	3.69	7.18	4.02
	Radio Young (2025)	6.45	5.52	4.65	4.03	6.04	5.25	3.72	7.30	4.08
	LRQ Lee et al. (2025)	6.46	5.54	4.67	4.05	6.48	5.57	3.89	7.12	4.12
	PDGQ	**5.75**	**5.25**	**4.24**	**3.68**	**5.65**	**5.02**	**3.44**	**6.55**	**3.33**
2-bit	RTN	1.9e3	781.20	68.04	15.08	4.2e3	122.08	27.27	1.9e3	4.6e5
	AWQ Lin et al. (2024)	2.6e5	2.8e5	2.4e5	7.4e4	2.2e5	1.2e5	-	1.7e6	1.7e6
	GPTQ Frantar et al. (2023)	152.31	20.44	13.01	9.51	60.45	28.14	8.78	210.00	11.90
	QuIP Chee et al. (2023)	29.74	12.48	11.57	7.83	39.73	13.48	6.64	84.97	13.03
	PB-LLM Yuan et al. (2024)	24.61	17.73	12.65	7.85	25.37	49.81	NaN	44.12	11.68
	SliM-LLM Huang et al. (2025)	14.58	8.87	7.33	5.90	16.01	9.41	6.28	39.66	9.46
	CBQ Ding et al. (2025)	10.60	7.50	5.58	5.25	8.01	7.00	5.10	26.80	8.15
	SliderQuant Wang et al. (2026)	10.30	7.40	5.65	5.15	9.59	7.71	6.53	27.59	8.10
	PDGQ	**6.57**	**5.84**	**4.83**	**4.18**	**6.50**	**5.54**	**3.94**	**7.94**	**4.98**

Table 7: Perplexity of LLaMA family on WikiText2 dataset.

wins once $p_{\text{sparse}} > 1/7 \approx 14.3\%$. The effective bit-width is therefore

$$\bar{b} = b \cdot (1 - p_{\text{out}} - p_{\text{sparse}}) + 23 \cdot p_{\text{out}} + \min(7 \cdot p_{\text{sparse}},\ 1)\,, \tag{54}$$

and all effective-bit-width figures reported in this paper (Table 10) use $\bar{b}$.

G Appendix: Perplexity Comparison in LLaMA

This section shows the details of perplexity on Wikitext2 and C4 dataset.

H Appendix: Zeroshot Accuracy Comparison in LLaMA families

This section shows the details of zero-shot accuracy on 6 datasets, including PIQA, HellaSwag, WinoGrande, ARC-E, ARC-C, and BoolQ.

The zero-shot evaluation results further demonstrate that PDGQ preserves downstream reasoning capability under aggressive low-bit compression. Across all LLaMA families and benchmark suites, PDGQ consistently achieves the strongest or near-strongest accuracy on most benchmarks, with the largest gains emerging in the ultra-low-bit regime. Comprehensive numerical results are provided in Appendix H.

At 2-bit, where most methods degrade severely, PDGQ stays close to full precision – 82.86/82.39/78.06/55.38 on PIQA/HellaSwag/WinoGrande/ARC-C for LLaMA-3 70B, which is 96–99% of FP16 accuracy on the first three and 94% on ARC-C, and the same pattern on LLaMA-2 70B

#bits	Method	1-7B	1-13B	1-30B	1-65B	2-7B	2-13B	2-70B	3-8B	3-70B
16	-	7.08	6.61	5.98	5.62	6.97	6.46	5.52	9.22	6.85
4-bit	RTN	7.72	7.12	6.58	6.15	7.41	6.75	5.95	10.12	7.15
	AWQ Lin et al. (2024)	**7.18**	6.83	6.19	5.78	6.93	**6.44**	5.68	**9.42**	**6.74**
	GPTQ Frantar et al. (2023)	7.21	6.87	6.24	5.82	6.97	6.48	5.72	9.55	6.82
	CBQ Ding et al. (2025)	-	-	**6.03**	**5.62**	7.05	-	-	-	-
	SliderQuant Wang et al. (2026)	-	-	-	-	7.19	6.54	**5.60**	9.94	-
	Radio Young (2025)	-	-	-	-	7.40	6.83	5.76	-	-
	PDGQ	7.59	**6.81**	6.15	5.83	**6.85**	6.74	5.74	9.61	7.38
3-bit	RTN	8.62	7.49	6.58	6.10	8.40	7.18	6.02	1.1e2	22.39
	AWQ Lin et al. (2024)	7.92	7.07	6.37	5.94	7.84	6.94	-	11.62	8.03
	GPTQ Frantar et al. (2023)	7.85	7.10	6.47	6.00	7.89	7.00	5.85	13.67	10.52
	SliM-LLM Huang et al. (2025)	**6.14**	**6.05**	6.33	5.94	7.74	**5.26**	**5.09**	13.10	8.64
	CBQ Ding et al. (2025)	-	-	-	-	**7.56**	-	-	-	-
	PDGQ	7.81	6.94	**6.26**	**5.92**	7.76	6.83	5.83	**10.15**	**7.68**
2-bit	RTN	1.0e3	4.5e2	99.45	17.15	4.9e3	1.4e2	42.13	2.5e4	4.6e5
	AWQ Lin et al. (2024)	1.9e5	2.3e5	2.4e5	7.5e4	1.7e5	9.4e4	-	2.1e6	1.4e6
	GPTQ Frantar et al. (2023)	34.63	15.29	11.93	11.99	33.70	20.97	-	4.1e4	21.82
	QuIP Chee et al. (2023)	33.74	21.94	10.95	13.99	31.94	16.16	8.17	1.3e2	22.24
	PB-LLM Yuan et al. (2024)	49.73	26.93	17.93	11.85	29.84	19.82	8.95	79.21	33.91
	SliM-LLM Huang et al. (2025)	32.91	13.85	11.27	10.95	16.00	9.41	7.01	1.1e2	15.92
	CBQ Ding et al. (2025)	-	-	7.65	7.42	11.30	-	-	-	-
	SliderQuant Wang et al. (2026)	-	-	-	-	13.83	11.21	9.59	56.98	-
	Radio Young (2025)	-	-	-	-	**8.04**	**7.22**	**5.99**	-	-
	PDGQ	**9.39**	**7.71**	**6.86**	**6.37**	9.17	7.30	6.31	**12.50**	**9.20**

Table 8: Perplexity of LLaMA family on C4 dataset. SliM-LLM results are quoted from Table 10 of Huang et al. (2025) rather than re-measured by us, and some (e.g. 3-bit on LLaMA-2 13B) fall below the FP16 row as reported there.

against GPTQ, AWQ, and QuIP. The effect is even sharper at smaller scales: on LLaMA-1 30B, PDGQ lifts HellaSwag from 71.55 (QuIP) and 66.10 (SliM-LLM) to 79.39, and on LLaMA-3 8B it retains 90–98% of FP16 accuracy at 2 bits and 95–101% at 3 bits while GPTQ and AWQ degrade substantially on commonsense reasoning. At 3-bit and 4-bit the gap narrows, but PDGQ remains competitive or best, particularly on reasoning-heavy benchmarks (ARC-C, HellaSwag, BoolQ), occasionally matching or exceeding full-precision accuracy.

Importantly, PDGQ's advantage consistently widens as bit-width decreases, supporting our central hypothesis that information-theoretic guidance becomes increasingly critical where magnitude-driven heuristics fail to distinguish tolerable noise from functionally destructive errors.

I Appendix: Qwen3

Full-family perplexity on Qwen3-Base. Table 2 extends the main-text LLaMA perplexity comparison (Section 5.2) to the entire Qwen3-Base family (0.6B–14B) against five baselines, including two recent (2025–2026) methods, SliderQuant and Radio, on both WikiText2 and C4. At 4-bit, GPTQ is marginally ahead of PDGQ on most sizes (e.g. a tie at 6.7 on 14B WikiText2, and a $\leq$0.3-point GPTQ edge elsewhere), with PDGQ only clearly ahead at the smallest, 0.6B model – but PDGQ still beats both SliderQuant and Radio at every size and dataset even here, so the only baseline it trails at 4-bit is the oldest one in the comparison. This reverses sharply as the bit budget shrinks: at 3-bit, AWQ has already begun to degrade severely (e.g. 85.9 at 0.6B WikiText2) while SliderQuant and Radio remain usable but consistently behind PDGQ (e.g. 20.0 and 21.5 vs. PDGQ's 17.5 at 0.6B WikiText2), which is best on all 10 cells; at 2-bit, AWQ and GPTQ collapse by several orders of magnitude (up to 10^8 and 10^3–10^4 respectively) while SliderQuant

and Radio degrade more gracefully but still trail PDGQ by a wide margin at every size (e.g. 40.0 and 46.0 vs. PDGQ's 30.5 at 0.6B WikiText2), and PDGQ is again best on all 10 cells. This closely mirrors the LLaMA results: PDGQ's advantage over the strongest baseline is negligible to slightly negative at 4-bit and grows to be decisive precisely in the ultra-low-bit regime the paper targets, on a model family the method was not designed or tuned against and against baselines as recent as 2026.

J Appendix: Component Analysis - LLaMA

To analyze the contribution of each component in PDGQ, we perform a comprehensive ablation study on LLaMA-2 7B with a group size of 128. Table 10 reports WikiText2 perplexity, the proportion of restored outliers, achieved sparsity, and effective average bit-width as we progressively enable activation weighting in the codebook (PDCodeOpt), Projected-Distortion Outlier Restoration (PDOR), and Distortion-Consistent Pruning (DCP). Every row uses the maximum-entropy quantile initialization, and the first row uses unweighted Lloyd updates.

Activation weighting in the codebook. All rows share the same maximum-entropy quantile initialization, so the drop from the first to the second row isolates the effect of weighting Lloyd's assignments and centroid updates by activation energy. In the 2-bit and 3-bit settings the few available codewords ($K = 2^b$) make every bit critical, and weighting places them where output distortion is most sensitive rather than where weights are most numerous. The effect of the initialization itself is analysed in Appendix L.

Selective Restoration via Projected-Distortion (PDOR). The effectiveness of PDOR highlights a crucial distinction between weight magnitude and functional sensitivity. While traditional outlier detection often relies on fixed thresholds or Hessian-based approximations, PDOR's activation-aware test statistic directly measures the "projected" error on the output manifold. Our observations show that restoring roughly 1% of weights (0.93% at 2 bits, 1.47% at 3 bits) closes 36% and 61% of the perplexity gap between the 2- and 3-bit codebooks and the 4-bit codebook. This implies that distortion is not uniformly distributed; rather, it is concentrated in a sparse set of "influence-heavy" weights. PDOR successfully identifies these weights, demonstrating that targeted high-precision restoration is more bit-efficient than increasing the global bit-width.

DCP as an Adaptive Rate-Distortion Optimizer. DCP serves as a unique self-correcting mechanism within the quantization pipeline. In standard quantization, every weight is forced into a bin, even if the quantization error is larger than the weight's original magnitude. DCP identifies these "harmful" assignments—where $\text{dist}(0) < \text{dist}(\hat{w})$—and recovers the wasted rate by pruning them to zero. Under an honest accounting that also charges the cost of recording *which* positions are pruned (Appendix F), this sparsity (up to 16.66% at 2-bit) does *not* reduce the effective bit-width relative to PDOR alone – it raises it (e.g., $2.20 \rightarrow 2.86$ bits at 2-bit), since the position bitmask needed once sparsity exceeds $\approx 14\%$ costs more than the low-bit codes it replaces. DCP's value is therefore not compression but quality: it removes weights whose quantized value actively increases distortion, delivering a substantial perplexity improvement (e.g., $7.55 \rightarrow 6.50$ at 2-bit) for a modest additional storage cost, and it does so without any hyperparameter tuning (e.g., manually setting sparsity ratios), making DCP a robust, plug-and-play component for diverse model architectures.

The cumulative improvements shown in Table 10 indicate that PDGQ components are not merely additive but synergistic. activation-weighted codebook optimization establishes a high-fidelity baseline; PDOR then "rescues" the outliers that the codebook cannot represent; finally, DCP "trims" the weights that the codebook represents poorly.

K APPENDIX: COMPONENT ANALYSIS - QWEN3

To confirm the component-analysis findings above are not specific to the LLaMA-2 architecture, and to isolate PDGQ's codebook construction and outlier-restoration criterion from comparable baselines under matched conditions (identical codebook optimization and DCP, only the compared component differs), we report additional studies on the Qwen3-Base family Yang et al. (2025) and LLaMA-2-7B.

Component analysis on Qwen3-8B-Base. Table 11 repeats the codebook-only / +outlier-restoration / +pruning / +both build-up of Table 10 on a second, more recent model family. The same pattern holds at every bit-width: outlier restoration and pruning each independently reduce perplexity relative to codebook-only, and combining both yields the lowest perplexity, confirming the two components are synergistic rather than an artifact of the LLaMA-2 architecture.

Bit-width	Variant	PPL	Restored-outlier %	Sparsity %
2-bit	Codebook-only	32.37	0.00%	0.00%
	+Outlier restoration	17.08	1.08%	0.00%
	+Pruning	28.64	0.00%	8.84%
	+Both (full PDGQ)	**13.41**	1.08%	8.84%
3-bit	Codebook-only	11.68	0.00%	0.00%
	+Outlier restoration	10.47	0.71%	0.00%
	+Pruning	11.12	0.00%	4.37%
	+Both (full PDGQ)	**10.08**	0.71%	4.37%
4-bit	Codebook-only	7.74	0.00%	0.00%
	+Outlier restoration	7.46	0.42%	0.00%
	+Pruning	7.57	0.00%	1.86%
	+Both (full PDGQ)	**7.31**	0.42%	1.86%

Table 11: Component-analysis ablation on Qwen3-8B-Base Yang et al. (2025) (WikiText2 perplexity).

Codebook optimization alone vs. QuIP. To isolate the codebook-construction contribution (Section 4.1) from outlier restoration and pruning, Table 12 compares PDGQ with PDOR and DCP both disabled against QuIP Chee et al. (2023) on LLaMA-2-7B WikiText2 perplexity. QuIP numbers are taken from the QuIP authors' own comment on OpenReview rather than the original paper table, since that is where this exact LLaMA-2-7B/WikiText2 configuration is reported.

Model	Bit-width	PDGQ (codebook-only)	QuIP
LLaMA-2-7B	2-bit	**8.68**	27.13
	3-bit	**6.13**	6.50
	4-bit	**5.57**	5.94

Table 12: Codebook-optimization-only (PDOR and DCP disabled) vs. QuIP Chee et al. (2023) on WikiText2 perplexity, LLaMA-2-7B.

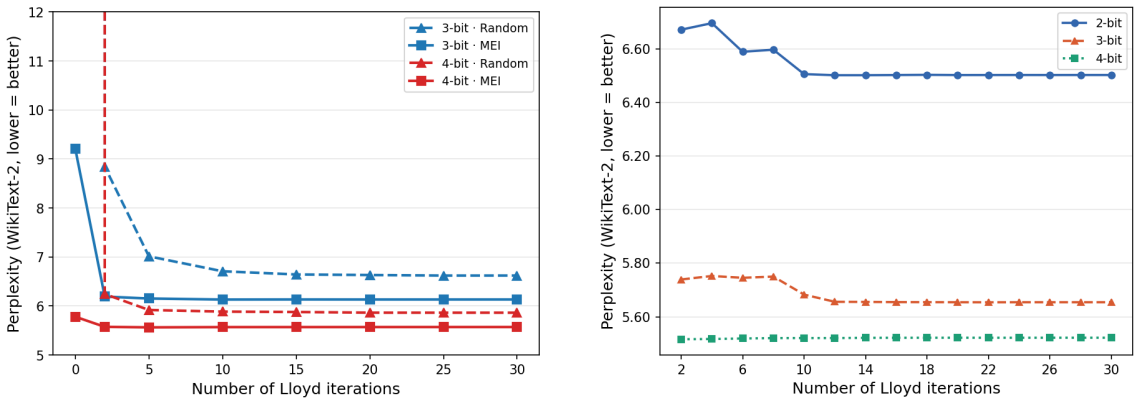
Outlier-restoration criterion alone vs. SqueezeLLM. Table 13 isolates the outlier-selection criterion by itself: codebook optimization and DCP are identical between rows, and only the restoration rule differs — PDOR's $\chi^2(1)$ test (Section 4.2) vs. a SqueezeLLM-style rule Kim et al. (2023) that flags weight (i, j)

as an outlier when it falls outside mean $\pm 3\sigma$ of its row-group's raw weight distribution. PDOR achieves lower perplexity than the SqueezeLLM criterion at every setting on both models, generally while restoring a larger fraction of weights, indicating that testing the residual against the rate-distortion noise floor $D^*(b, g)$ identifies more functionally relevant outliers than a fixed statistical threshold on the raw weight alone.

Model	Bit-width	Criterion	PPL	Outlier %
LLaMA-2-7B	2-bit	SqueezeLLM (mean $\pm 3\sigma$)	6.92	0.26%
		PDOR (ours)	**6.50**	0.93%
	3-bit	SqueezeLLM (mean $\pm 3\sigma$)	5.93	0.26%
		PDOR (ours)	**5.65**	1.47%
Qwen3-8B-Base	2-bit	SqueezeLLM (mean $\pm 3\sigma$)	31.90	0.29%
		PDOR (ours)	**13.41**	1.08%
	3-bit	SqueezeLLM (mean $\pm 3\sigma$)	16.80	0.29%
		PDOR (ours)	**10.08**	0.71%

Table 13: Outlier-restoration criterion alone: PDOR vs. a SqueezeLLM-style Kim et al. (2023) mean$\pm 3\sigma$ rule, with codebook optimization and DCP held identical (WikiText2 perplexity).

L Convergence Analysis of Activation-Aware Lloyd Refinement



(a) Effect of Maximum-Entropy Initialization on perplexity 3- and 4-bit on LLaMA-2 7B

(b) Effect of Lloyd iterations on perplexity 2-, 3-, and 4-bit on LLaMA-2 7B

Figure 2: Effect of Maximum Entropy Quantization on performance at different bit-widths for LLaMA-2 7B.

MEQ Initialization Advantage. Figure 2a clearly demonstrates the benefit of Maximum-Entropy initialization over random initialization. Random initialization begins at substantially higher perplexity — approximately 9.2 for 3-bit and 6.1 for 4-bit at iteration 0 — and requires many more iterations to converge. In contrast, MEQ initialization starts from a markedly lower perplexity baseline (approximately 5.8 for 3-bit and 5.6 for 4-bit) and converges within as few as 5 Lloyd iterations. This gap confirms that maximum-

entropy initialization provides a significantly better starting point for the optimization landscape, reducing the risk of centroid collapse and accelerating convergence.

Convergence Behavior of Activation-Aware Lloyd Refinement. Figure 2b illustrates the convergence behavior of our activation-aware Lloyd refinement. Unlike standard Lloyd quantization, which minimizes raw ℓ_2 weight distortion, our refinement is guided by the projected error on activations, creating a more informed optimization landscape where centroids reach a high-fidelity state rapidly. The steep perplexity improvement in the first few iterations suggests that the codebook quickly captures the most activation-influential features of the weight distribution. The number of iterations required for stability correlates with bit-width: the 2-bit regime continues to improve noticeably up to $J \approx 12$ due to its highly constrained search space ($K = 4$ centroids), whereas 3-bit and 4-bit configurations plateau earlier around $J \approx 8$–10. Beyond $J = 16$, marginal gains become negligible across all bit-widths, with the 4-bit regime being largely insensitive to iteration count given its richer codebook ($K = 16$). We set $J = 20$ for all main experiments as a convergence safety margin that ensures stability across diverse layer types without significantly increasing calibration overhead.

M LIMITATIONS

While PDGQ demonstrates consistent improvements across the LLaMA model family, several limitations warrant discussion.

Gaussian Weight Assumption. The rate-distortion lower bound $D^*(b) = \sigma_w^2 \cdot 2^{-2b}$ used in PDOR's test statistic is derived under the assumption that weights are approximately Gaussian. Appendix D.3.11 formalizes exactly what this assumption buys (the null hypothesis is the scalar specialization of the optimal rate-distortion test channel, not an arbitrary noise model) and quantifies its real-world gap empirically: the observed test statistic's median exceeds the theoretical $\chi^2(1)$ median by $1.7\times$ at b=2 and on the layer measured the true flagging rate runs roughly double the nominal α. In practice, LLM weights exhibit heavier-tailed distributions (e.g., Laplace or Student-t), which means the bound serves as a conservative baseline. While this makes the restoration criterion robust rather than brittle, the realized restoration rate can therefore differ from α in either direction, and the criterion may over- or under-restore outliers in layers with strongly non-Gaussian weight distributions.

Diagonal Activation Covariance Approximation. The projected-distortion objective keeps only the diagonal of the activation covariance and therefore ignores cross-channel correlations. Appendix D.2 measures the resulting error on Qwen2.5-0.5B-Instruct and DeepSeek-R1-Distill-Qwen-1.5B. With $\mathbb{E}[x_i^2]$ on the diagonal, the estimate is within 1% of the exact distortion at the median, but it underestimates it by up to 30% in individual first and last layers at 2 bits, and the centered variance PDGQ uses is about 10% lower still. Replacing the diagonal by block- or full-covariance objectives improves the codebook stage substantially but not the final perplexity of the complete pipeline (Appendix D.2); this was tested with a local solver on models of at most 1.5B parameters, and we have not evaluated it at 70B scale.

Calibration Data Sensitivity. Like all activation-aware PTQ methods, PDGQ relies on a small calibration set (256 samples from C4) to estimate $\sigma_x^2(i)$. The quality of these estimates directly affects both the codebook refinement and the outlier restoration decisions. Calibration data that is poorly representative of the deployment distribution may lead to miscalibrated activation statistics, degrading the advantage of the activation-aware objective. A systematic study of calibration data size and domain mismatch is left for future work.

Memory Overhead of Outlier Storage. Restored outliers are stored as full-precision FP16 scalars, and both restored and pruned positions require recording *which* weights they are – a cost Appendix F prices exactly (a 7-bit sparse index per position within a 128-wide group, or a 1-bit-per-weight bitmask when pruning exceeds $\approx$14% of a group, whichever is cheaper), and the effective bit-widths reported in the component analysis include it; notably, this shows DCP's sparsity is not free compression once indexing is included (Appendix F), only a favorable perplexity-for-bits trade. What remains open is efficient hardware support for the resulting mixed-precision layout (quantized codewords, FP16 scalars, and a position index or bitmask, interleaved per row-group) – implementing and benchmarking a dedicated inference kernel for this representation is left to future work.

Scope of Evaluation. Our main experiments are conducted on the LLaMA model family (7B–70B). Appendix K additionally evaluates the full Qwen3-Base family (0.6B–14B) and Section 5.4 evaluates DeepSeek-R1-Distill-Qwen reasoning models (1.5B–14B) – both structurally distinct from LLaMA – and PDGQ's advantage over baselines holds in both cases, particularly in the ultra-low-bit regime. Generalization to architectures with qualitatively different components, such as mixture-of-experts layers (e.g. Mixtral) or multimodal models, remains empirically unverified, and such architectures may still require re-tuning of the PDOR significance level α or the Lloyd iteration count J.

Sequential Layer-wise Processing. Following standard PTQ practice, PDGQ processes layers sequentially and does not account for error interactions across layers. Quantization decisions made for an early layer affect the activation distribution seen by later layers, but this feedback is not incorporated into the optimization. Joint or block-wise optimization could potentially reduce accumulated error, at the cost of increased calibration complexity.

Model	#bits	Method	PIQA	HellaS	WinoG	ARC-E	ARC-C	BoolQ
L-7	16	Original	79.11	75.13	70.32	67.47	41.72	76.70
	2	GPTQ	55.39	27.36	48.30	30.64	19.62	44.74
		AWQ	52.83	25.27	49.88	25.25	22.44	37.83
		QuIP	67.52	53.21	60.84	49.15	32.42	59.50
		Slim-LLM	57.83	36.70	56.24	33.46	25.09	56.05
		PDGQ	**77.09**	**71.85**	**68.59**	**61.03**	**39.25**	**74.22**
	3	GPTQ	77.12	72.54	67.45	65.21	38.92	73.15
		AWQ	77.89	73.41	68.22	66.08	40.15	75.28
		QuIP	**78.31**	**74.22**	69.10	67.01	**41.03**	76.04
		SliderQuant	77.42	71.74	67.96	**69.70**	40.36	—
		PDGQ	78.29	74.14	**69.77**	65.91	40.87	**76.24**
	4	GPTQ	77.08	74.32	68.75	65.66	39.51	71.35
		AWQ	78.46	74.45	68.43	66.50	40.78	73.55
		QuIP	**78.84**	**74.82**	69.91	67.18	41.53	**76.48**
		PDGQ	78.07	74.71	**70.32**	**67.34**	**41.81**	76.42
L-13	16	Original	80.10	79.20	73.00	74.80	47.40	78.10
	2	GPTQ	54.12	28.36	51.20	32.40	21.55	48.20
		AWQ	65.48	44.15	58.70	45.22	28.90	56.40
		QuIP	74.25	64.88	63.50	**66.14**	36.50	70.30
		SliM-LLM	73.19	63.04	61.79	47.95	36.27	55.92
		PDGQ	**78.35**	**75.78**	**70.17**	65.91	**42.58**	**78.10**
	3	GPTQ	78.45	76.10	71.20	68.30	43.15	75.40
		AWQ	79.12	77.45	71.90	70.55	44.80	76.90
		QuIP	79.60	**78.52**	72.40	**73.20**	**46.10**	77.50
		PDGQ	**79.71**	77.73	**72.77**	68.35	45.90	**78.38**
	4	GPTQ	79.45	78.50	72.10	72.30	45.80	77.20
		AWQ	79.70	78.75	72.45	73.10	46.20	77.90
		QuIP	**79.92**	**78.95**	**72.85**	**74.52**	**47.10**	77.95
		PDGQ	79.82	78.25	72.77	68.01	46.33	**79.85**
L-30	16	Original	81.30	82.80	76.80	76.60	50.10	82.50
	2	GPTQ	59.20	35.10	50.40	38.60	22.40	51.10
		AWQ	68.35	48.92	59.40	47.12	31.45	58.90
		QuIP	78.42	71.55	70.12	65.80	42.10	74.35
		CBQ	77.23	75.05	—	57.12	42.93	—
		SliM-LLM	75.52	66.10	64.07	51.29	39.29	62.01
		PDGQ	**80.14**	**79.39**	**75.06**	**72.05**	**50.34**	**81.25**
	3	GPTQ	79.80	80.45	74.80	73.50	47.90	80.60
		AWQ	80.12	81.05	75.30	74.15	48.20	81.40
		QuIP	80.95	**82.20**	**76.40**	**75.95**	49.65	82.10
		PDGQ	**81.18**	81.38	76.16	73.78	**49.83**	**83.46**
	4	GPTQ	80.85	82.10	75.90	75.20	48.90	81.80
		AWQ	81.10	82.35	76.20	75.80	49.50	82.15
		QuIP	81.15	**82.55**	**76.45**	**76.32**	49.80	82.25
		CBQ	80.12	79.11	—	59.89	46.65	—
		PDGQ	**81.28**	81.58	75.37	74.58	**50.68**	**83.58**
L-65	16	Original	82.10	83.21	77.27	73.48	51.45	85.57
	2	GPTQ	68.45	49.20	58.10	45.30	30.50	62.40
		AWQ	71.20	55.40	63.85	51.45	34.20	65.10
		QuIP	79.85	76.40	73.90	68.10	45.35	79.20
		CBQ	78.12	74.28	—	55.35	41.64	—
		SliM-LLM	77.09	72.05	70.91	53.72	40.25	77.51
		PDGQ	**81.28**	**81.95**	**76.40**	**69.53**	**48.29**	**83.18**
	3	GPTQ	81.25	82.30	76.50	72.85	50.15	84.80
		AWQ	81.65	82.75	76.85	73.10	50.80	85.10
		QuIP	82.05	**83.15**	77.40	73.65	**51.25**	85.70
		PDGQ	**82.10**	83.11	**77.58**	**73.91**	51.11	**85.84**
	4	GPTQ	81.90	83.05	77.15	73.30	51.35	85.45
		AWQ	82.05	83.10	**77.22**	**73.40**	51.38	85.50
		QuIP	81.95	83.05	77.10	73.25	51.30	85.35
		CBQ	81.12	80.76	—	58.64	45.98	—
		PDGQ	**82.15**	**83.30**	76.72	73.32	**51.54**	**85.57**

(a) LLaMA-1 family

Model	#bits	Method	PIQA	HellaS	WinoG	ARC-E	ARC-C	BoolQ
L2-7	16	Original	78.78	75.00	69.53	67.21	42.58	79.30
	2	GPTQ	58.05	32.65	49.88	33.16	21.42	55.47
		AWQ	52.39	25.58	49.09	21.16	25.58	37.83
		QuIP	69.45	55.10	62.30	50.80	33.45	62.10
		SliM-LLM	63.82	33.76	56.91	47.81	23.38	59.97
		SliderQuant	70.78	57.15	60.14	57.79	31.06	66.12
		CBQ	71.59	60.28	—	45.74	32.93	—
		PDGQ	**77.04**	**72.28**	**65.35**	**61.03**	**39.85**	**74.77**
	3	GPTQ	77.45	72.10	66.85	64.90	40.20	74.85
		AWQ	78.20	73.15	67.50	66.10	41.85	76.20
		QuIP	78.40	73.85	68.10	66.50	42.15	**76.90**
		CBQ	76.25	71.34	—	52.36	39.21	—
		PDGQ	**78.56**	**74.41**	**69.14**	**67.05**	**42.32**	76.88
	4	GPTQ	78.85	74.65	68.70	67.45	43.10	77.80
		AWQ	**79.05**	74.90	68.95	68.00	43.55	78.15
		QuIP	78.65	74.85	69.30	67.05	42.35	**79.10**
		SliderQuant	78.67	**75.90**	**69.46**	**74.62**	**46.42**	78.01
		CBQ	77.34	72.23	—	53.66	40.22	—
		PDGQ	78.73	74.69	68.75	67.59	42.83	78.32
L2-13	16	Original	80.69	78.66	72.61	71.42	46.67	82.11
	2	GPTQ	65.40	42.15	55.30	48.20	28.15	58.40
		AWQ	66.85	45.30	59.20	46.50	30.15	57.80
		QuIP	75.10	66.25	64.15	66.90	37.80	71.45
		SliderQuant	73.56	64.75	63.22	67.47	37.54	70.67
		PDGQ	**79.65**	**76.61**	**72.14**	**70.12**	**45.65**	**79.17**
	3	GPTQ	76.80	68.40	66.20	64.30	41.50	72.10
		AWQ	79.55	77.20	71.40	70.25	45.10	80.50
		QuIP	**79.95**	**78.10**	**72.15**	**70.80**	45.90	**81.35**
		PDGQ	79.92	78.06	71.98	70.71	**47.18**	81.25
	4	GPTQ	80.12	78.15	71.85	71.05	46.22	81.75
		AWQ	80.45	78.38	72.20	71.25	46.50	**82.00**
		QuIP	80.48	**78.45**	**72.40**	71.25	46.45	81.90
		SliderQuant	80.41	78.30	72.14	**76.73**	**48.55**	78.10
		PDGQ	**80.69**	78.43	71.98	71.09	47.27	81.90
L2-70	16	Original	82.05	83.38	80.51	74.54	53.33	85.17
	2	GPTQ	72.10	58.45	64.20	60.15	38.90	67.30
		AWQ	75.30	66.80	72.40	64.20	44.15	74.80
		QuIP	80.45	78.10	76.55	70.20	48.90	81.25
		PDGQ	**81.18**	**82.53**	**79.01**	**72.64**	**52.73**	**83.76**
	3	GPTQ	80.50	81.20	78.40	72.80	51.40	82.60
		AWQ	81.95	82.70	79.85	74.50	53.10	84.45
		QuIP	82.10	83.05	**80.05**	**74.85**	53.40	**84.75**
		PDGQ	**82.26**	**83.11**	79.56	74.16	**53.67**	83.98
	4	GPTQ	81.90	83.15	80.10	74.20	53.10	84.95
		AWQ	82.00	**83.30**	**80.40**	**74.45**	**53.25**	**85.10**
		QuIP	82.00	83.25	80.35	74.40	53.20	84.95
		PDGQ	**82.32**	83.27	79.72	74.16	53.24	84.83

(b) LLaMA-2 family

Model	#bits	Method	PIQA	HellaS	WinoG	ARC-E	ARC-C	BoolQ
L3-8	16	Original	79.71	78.81	73.72	76.43	53.33	82.29
	2	GPTQ	52.12	26.72	49.17	25.72	21.59	44.16
		AWQ	52.72	25.58	49.33	24.16	21.50	62.17
		QuIP	68.20	54.40	62.10	58.50	36.40	65.20
		Slim-LLM	55.77	27.83	48.78	32.15	19.11	44.83
		PDGQ	**78.45**	**74.89**	**72.14**	**73.57**	**47.78**	**76.06**
	3	GPTQ	74.90	54.30	71.10	70.50	37.70	—
		AWQ	77.70	55.10	72.10	74.00	43.20	—
		QuIP	76.50	73.10	70.80	72.40	48.60	78.40
		Slim-LLM	77.80	55.50	72.80	73.70	42.90	—
		PDGQ	**79.54**	**78.22**	**73.95**	**76.98**	**50.85**	**80.52**
	4	GPTQ	78.40	59.00	72.60	78.80	47.70	—
		AWQ	79.10	59.10	**74.00**	79.70	49.30	—
		QuIP	78.20	58.60	73.20	78.20	47.40	—
		Slim-LLM	78.90	58.70	72.60	**79.90**	49.40	—
		PDGQ	**79.87**	**78.47**	73.01	75.55	**52.65**	**81.77**
L3-70	16	Original	83.57	85.05	81.22	77.82	59.04	86.79
	2	GPTQ	74.30	62.10	65.50	63.20	41.50	69.80
		AWQ	76.45	64.20	68.30	65.10	43.25	71.50
		QuIP	81.10	79.55	77.40	72.85	50.10	81.90
		PDGQ	**82.86**	**82.39**	**78.06**	**76.39**	**55.38**	**82.23**
	3	GPTQ	81.90	82.80	78.90	76.10	56.40	84.20
		AWQ	82.80	83.90	80.15	76.50	57.10	85.55
		QuIP	**83.15**	84.40	**80.75**	77.20	58.35	86.20
		PDGQ	82.75	**84.50**	80.11	**78.62**	**59.39**	**86.36**
	4	GPTQ	83.35	84.85	80.95	77.55	58.70	86.55
		AWQ	**83.45**	84.95	**81.10**	77.70	58.90	86.70
		QuIP	**83.45**	84.95	81.05	77.65	58.85	86.60
		PDGQ	83.41	**84.99**	80.35	**77.95**	**59.90**	**86.82**

(c) LLaMA-3 family

Table 9: Zeroshot Accuracy (↑) of LLaMA family on 6 datasets. **Bold** = best; underline = second best. "—" denotes missing results.

#bits	PDCodeOpt	PDOR		DCP	WikiText2	Outlier	Sparsity	AVG
	Act. Weight	Use	Act. Weight		(↓)	(↓)	(↑)	Bits
2	-	-	-	-	10.5842	0.00	0.00	2.00
	✓	-	-	-	8.6827	0.00	0.00	2.00
	✓	✓	-	-	7.6028	1.13%	0.00	2.24
	✓	✓	✓	-	7.5513	0.93%	0.00	2.20
	✓	✓	✓	✓	**6.50**	0.93%	16.66%	2.86
3	-	-	-	-	7.1302	0.00	0.00	3.00
	✓	-	-	-	6.1319	0.00	0.00	3.00
	✓	✓	-	-	5.7965	1.81%	0.00	3.37
	✓	✓	✓	-	5.7911	1.47%	0.00	3.29
	✓	✓	✓	✓	**5.65**	1.47%	8.73%	3.64
4	-	-	-	-	5.6665	0.00	0.00	4.00
	✓	-	-	-	5.5696	0.00	0.00	4.00
	✓	✓	-	-	5.5384	1.34%	0.00	4.25
	✓	✓	✓	-	5.5350	1.07%	0.00	4.20
	✓	✓	✓	✓	**5.52**	1.07%	3.80%	4.32

Table 10: Effect of PDGQ components (Projected-Distortion Codebook Optimization (PDCodeOpt), Per-weight Projected-Distortion for Outlier Restoration (PDOR), and Distortion-Consistent Pruning (DCP)) on LLaMA-2 7B. All rows use the maximum-entropy quantile initialization, and the first row uses unweighted Lloyd updates.