PRISMQ: NOISE-AWARE RESOURCE ALLOCATION FOR ELASTIC POST-TRAINING QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Elastic post-training quantization (PTQ) jointly calibrates multiple precisions within a single model, enabling flexible precision adaptation under different deployment budgets. However, extending reconstruction from a single precision to multiple precisions introduces two key challenges. First, quantization errors vary across linear layers and precisions, creating a trade-off between cross-precision parameter sharing and precision-specific reconstruction capacity within the reconstruction space. Second, calibration states should reflect precision-dependent error propagation, while preserving complete quantized states for every precision incurs substantial calibration memory overhead. To address these challenges, we propose PrismQ, a noise-aware elastic PTQ framework that adapts both reconstruction space and calibration states according to precision-dependent quantization errors. Specifically, PrismRank estimates the demand for reconstruction capacity at each linear layer and precision from the singular-value characteristics of clipping-induced errors, and PrismLoRA realizes the resulting capacity allocation through shared and precision-specific low-rank components. Moreover, Calibration-only Sparse Token selects quantized token states according to token-wise quantization error under a fixed cache budget and uses them to construct calibration inputs for the next block, allowing these inputs to reflect precision-dependent error propagation. Extensive experiments on language models, vision transformers, and vision-language models show that PrismQ improves quantization performance across multiple precisions, with particularly clear gains at low bit widths.

1 INTRODUCTION

Post-training quantization (PTQ) has become a practical approach for reducing the computation and memory costs of modern neural networks without expensive retraining. However, conventional PTQ methods Frantar et al. (2023); Shao et al. (2024); Ashkboos et al. (2024); Lin et al. (2024a) typically optimize each target precision independently, leading to repeated calibration when models need to operate under different deployment budgets. Elastic PTQ Yu et al. (2021); Shkolnik et al. (2020); Xu et al. (2023); Park et al. (2024); Nair et al. (2025); Xu et al. (2024; 2026b) addresses this limitation by jointly calibrating multiple bit widths within a single model, enabling flexible precision adaptation while avoiding repeated optimization for individual precisions.

Extending reconstruction from a single precision to multiple precisions, however, introduces two key trade-offs. First, the reconstruction space faces a trade-off between cross-precision parameter sharing and precision-specific reconstruction capacity. Existing approaches commonly reuse parameters across bit widths through nested representations or cascaded low-rank structures, which improves parameter efficiency but constrains different precisions to a largely predefined sharing pattern (Nair et al., 2025; Xu et al., 2026b). Since quantization errors can vary substantially across linear layers and precisions, such predefined sharing may provide insufficient reconstruction capacity for precision settings with larger errors, whereas assigning independent reconstruction capacity to each precision increases redundancy. Second, calibration states face a trade-off between representing precision-dependent input errors and controlling calibration overhead. Existing methods typically construct calibration inputs by uniformly mixing states from different bit widths or heuristically retaining and merging selected precision-specific states (Xu et al., 2024; 2026b). Although

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.

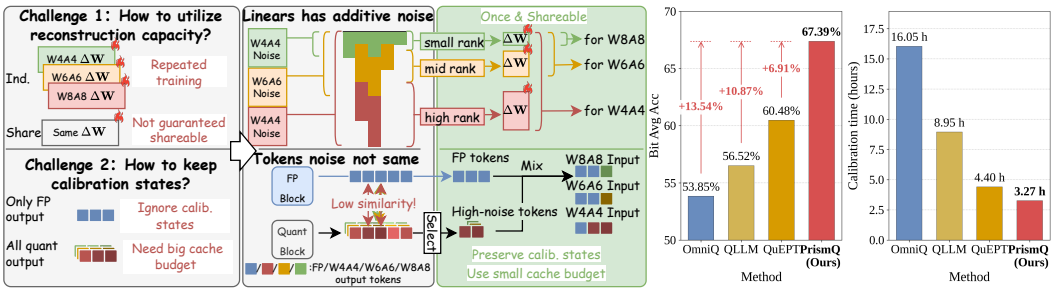


Figure 1: Motivation for noise-aware allocation of reconstruction space and calibration states. Left and center: precision-dependent quantization errors motivate allocating reconstruction capacity through shared and precision-specific low-rank components. Selected quantized token states serve as calibration states for the next block under a fixed cache budget, preserving precision-dependent quantization errors across successive block-wise calibration stages. Right: LLaMA-2-7B five-task accuracy averaged over W4A4 and W6A6, together with calibration time.

these strategies expose the next transformer block to multi-precision quantization effects, they do not explicitly determine which quantized states best represent the reconstruction demand at each precision. This raises a key question: *how can limited reconstruction capacity and calibration states be allocated according to the actual quantization errors of different bit widths?*

These two trade-offs suggest a unified resource-allocation view of elastic PTQ, as illustrated in Figure 1. Rather than relying on predefined parameter-sharing and calibration-state propagation strategies, we view multi-precision reconstruction as a noise-aware allocation problem over two limited resources: reconstruction capacity and calibration states. Reconstruction capacity determines how shared and precision-specific low-rank components are distributed across linear layers and precisions, while calibration-state allocation determines which quantized token states are preserved under a fixed cache budget and propagated as calibration inputs to the next transformer block.

Based on this perspective, we propose PrismQ, a noise-aware elastic PTQ framework that jointly adapts reconstruction space and calibration states. For reconstruction-space allocation, PrismRank estimates the demand for reconstruction capacity at each linear layer and precision from the singular-value characteristics of the clipping-induced error, while PrismLoRA realizes the resulting capacity allocation through shared and precision-specific low-rank components. For calibration-state allocation, Sparse Token (ST) selects quantized token states according to token-wise quantization error under a fixed cache budget and uses them to construct calibration inputs for the next transformer block, so that its calibration reflects precision-dependent error propagation without retaining complete quantized states for every candidate precision.

We evaluate PrismQ on large language models, vision transformers, and vision-language models across multiple precision settings. The results show improvements across multiple precisions, with larger gains under low-bit settings. As a representative example in Figure 1, on LLaMA-2-7B, PrismQ improves the five-task accuracy averaged over W4A4 and W6A6 by 6.91 percentage points over QuEPT, from 60.48% to 67.39%, while reducing the joint calibration time from 4.40 to 3.27 hours. Controlled ablations further examine the effects of quantization-error analysis, allocation scores, low-rank sharing structures, and calibration-state propagation.

Our contributions are summarized as follows:

- We frame elastic PTQ as a noise-aware resource allocation problem for multi-bit reconstruction, noting two trade-offs in reconstruction-space and calibration-state allocation.
- We propose PrismRank and PrismLoRA to estimate the reconstruction demand of each linear layer based on multi-bit-width quantization error, and realize demand-aware reconstruction spaces through shared and precision-specific low-rank components.
- We introduce Sparse Token to select and propagate multi-bit quantized token states under a fixed cache budget, and demonstrate the effectiveness of PrismQ across language, vision, and vision-language models through extensive experiments.

2 Related Work

2.1 Elastic precision quantization

Conventional PTQ methods improve low-bit accuracy through weight reconstruction or distribution transformation. Representative approaches include GPTQ (Frantar et al., 2023), which minimizes layer-wise reconstruction error, and AWQ (Lin et al., 2024b), SmoothQuant (Xiao et al., 2023), and OmniQuant (Shao et al., 2024), which reduce quantization difficulty through activation-aware scaling or learnable transformations. In contrast, elastic precision quantization (Yu et al., 2021; Xu et al., 2023; 2026b;a) aims to support multiple bit-width configurations within a single model or calibration process, avoiding repeated optimization for different precisions. Specifically, MBQuant (Zhong et al., 2025b) employs a multi-branch topology to support arbitrary bit-width quantization while alleviating errors caused by bit-width switching, whereas PTMQ (Xu et al., 2024) mixes features from different bit-width groups to construct calibration inputs and MatQuant (Nair et al., 2025) learns nested representations shared across precision levels. Although these approaches improve multi-precision deployment efficiency, their parameter-sharing and calibration strategies are largely predefined, without explicitly allocating quantization parameters according to layer-wise and precision-specific quantization errors.

2.2 LoRA-based quantization

Low-rank adaptation represents weight updates with two thin matrices (Hu et al., 2022) and has been widely incorporated into quantization to improve reconstruction efficiency. QLoRA (Dettmers et al., 2023), LoftQ (Li et al., 2024), LQ-LoRA (Guo et al., 2024), and LR-QAT (Bondarenko et al., 2024) integrate low-rank updates with quantized training or reconstruction, while LRQuant (Zhao et al., 2024), LRQuant+ (Zhao et al., 2025), ABQ-LLM (Zeng et al., 2024), and PEQuant (Xu et al., 2026a) further combine low-rank reconstruction with distribution transformations or feature correction. More recently, QuEPT (Xu et al., 2026b) and CoA-LoRA (Ye et al., 2026) extend low-rank reconstruction to elastic quantization by sharing recovery components across precisions or conditioning them on quantization configurations. However, these methods generally adopt predefined low-rank structures or capacities, without explicitly considering that reconstruction difficulty can vary substantially across linear layers and precision levels. In contrast, PrismQ estimates layer- and precision-specific reconstruction demand from clipping-induced quantization errors and adaptively allocates shared and precision-specific low-rank capacity accordingly.

3 Method

Figure 2 summarizes block-wise joint calibration. PrismRank allocates reconstruction capacity from clipping-induced errors; PrismLoRA realizes it through shared and precision-specific low-rank components. After joint reconstruction, Sparse Token selects quantized token states under a fixed cache budget and passes the resulting calibration states to the next transformer block.

3.1 Clipping-induced error and cross-precision structure

For a linear layer, let d_{in}, d_{out}, and n denote input dimension, output dimension, and tokens per sample. Full-precision weights and inputs are $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ and $\mathbf{X} \in \mathbb{R}^{d_{\text{in}} \times n}$. Let $b \in \mathcal{B}$ index deployment-specified weight–activation formats. Weight and activation clipping operators, identified by superscripts w and a, define $\mathbf{W}_b^c = \mathcal{C}_b^w(\mathbf{W})$ and $\mathbf{X}_b^c = \mathcal{C}_b^a(\mathbf{X})$, where c denotes clipped tensors. Before low-rank correction, disabling rounding gives the exact clipping-induced error decomposition

$$\mathbf{W}\mathbf{X} - \mathbf{W}_b^c\mathbf{X}_b^c = \underbrace{\mathbf{W}(\mathbf{X} - \mathbf{X}_b^c)}_{\text{activation clipping error}} + \underbrace{(\mathbf{W} - \mathbf{W}_b^c)\mathbf{X}_b^c}_{\text{weight clipping error}}. \tag{1}$$

Weight clipping error depends on clipped activations: algebraic separation does not imply statistical independence. Therefore, subsequent reconstruction jointly optimizes the weight and activation clipping errors.

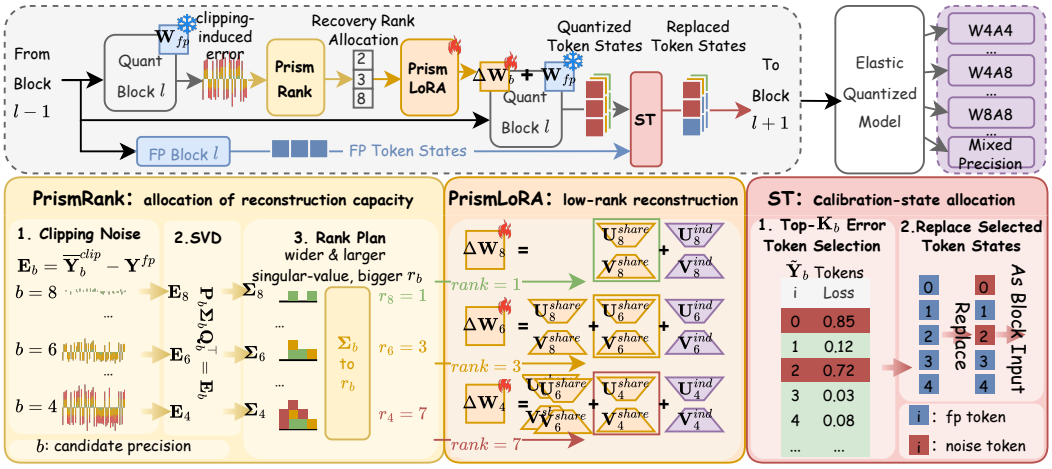


Figure 2: Overall framework of PrismQ. PrismRank estimates layer- and precision-specific reconstruction demand from clipping-induced errors, while PrismLoRA allocates shared and precision-specific low-rank components accordingly. Sparse Token further selects quantized token states under a fixed cache budget to construct calibration inputs for subsequent blocks. The calibrated model supports multiple weight–activation precision configurations.

Order $(b_1, \ldots, b_m)$ from lower to higher weight and activation precision, with b_i the ith candidate. For a fixed group of $\mathbf{T} \in \{\mathbf{W}, \mathbf{X}\}$, let $\mathcal{I}_{b_i}^{\mathbf{T}}$ denote its clipping interval. Under common grouping, nesting means

$$\mathcal{I}_{b_1}^{\mathbf{W}} \subseteq \cdots \subseteq \mathcal{I}_{b_m}^{\mathbf{W}}, \qquad \mathcal{I}_{b_1}^{\mathbf{X}} \subseteq \cdots \subseteq \mathcal{I}_{b_m}^{\mathbf{X}}. \tag{2}$$

For fixed weights and inputs, define $\mathbf{C}_{b_i}^{\mathbf{T}} = \mathbf{T} - \mathbf{T}_{b_i}^{c}$. Adjacent precisions satisfy

$$\mathbf{C}_{b_i}^{\mathbf{T}} = \mathbf{C}_{b_{i+1}}^{\mathbf{T}} + (\mathbf{T}_{b_{i+1}}^{c} - \mathbf{T}_{b_i}^{c}), \qquad \mathbf{T} \in \{\mathbf{W}, \mathbf{X}\}. \tag{3}$$

Both errors thus comprise a highest-precision base plus successive increments (Appendix H). Under Equation 2, increments represent additional clipping at lower precisions, motivating partial sharing of reconstruction parameters across precisions.

Three complementary sources support this interpretation of cross-precision clipping structure: a weight-distribution model, empirical weight-interval statistics, and conditional nesting of activation intervals. First, for a fixed weight group with a fixed Laplace distribution and a common clipping center, the approximate scalar quantization-MSE model predicts that the optimal clipping radius increases with bit width (Appendix H). This provides a theoretical basis for nested intervals within the model, without imposing nesting on independently learned bounds. Second, the dedicated LLaMA-2-7B check reports 76.29%–88.99% nesting across adjacent precision pairs after clipping pretraining, with per-output-channel quantization at every weight precision (Table 20). These measurements establish nesting as a common pattern at this stage, rather than a property of every interval or of bounds after joint reconstruction. Third, for the same activation input and grouping, clipping ratios that increase monotonically with bit width produce nested intervals when the min–max range contains zero: higher-precision intervals then contain lower-precision intervals. Together, these findings motivate shared low-rank components. Imperfect nesting and rounding error within clipping intervals also motivate precision-specific components, allowing each precision to retain reconstruction capacity beyond the shared structure.

3.2 PrismRank: Noise-Aware Allocation of Reconstruction Capacity

After clipping-bound pretraining and before joint block reconstruction, PrismRank disables rounding while retaining learned clipping bounds. At linear layer j in transformer block l and format $b \in \mathcal{B}$, full-precision output $\mathbf{Y}_{l,j}^{\mathrm{fp}}$ and clipping-only output $\overline{\mathbf{Y}}_{l,j,b}^{\mathrm{clip}}$ define

$$\mathbf{E}_{l,j,b} = \overline{\mathbf{Y}}_{l,j,b}^{\mathrm{clip}} - \mathbf{Y}_{l,j}^{\mathrm{fp}}. \tag{4}$$

This combined error estimates reconstruction demand, the demand for reconstruction capacity, without requiring Equation 2 at every group. Its sign reverses Equation 1, leaving singular values unchanged. For each of n_s rank-analysis samples, write $\mathbf{E}_b$ for the error with l, j, and the sample index suppressed. Its exact singular value decomposition is

$$\mathbf{P}_b \mathbf{\Sigma}_b \mathbf{Q}_b^\top = \mathbf{E}_b, \tag{5}$$

where $\mathbf{P}_b$ and $\mathbf{Q}_b$ contain left and right singular vectors, and diagonal $\mathbf{\Sigma}_b$ contains singular values $\sigma_1 \geq \sigma_2 \geq \cdots$. With aggregation exponent p and retention threshold τ, the retained rank is

$$r_\tau(\mathbf{E}) = \min \left\{ r : \frac{\sum_{i=1}^{r} |\sigma_i(\mathbf{E})|^p}{\sum_i |\sigma_i(\mathbf{E})|^p} \geq \tau \right\}. \tag{6}$$

Appendix G specifies p, τ, and clipping-pretraining and rank-analysis sample counts. The Power×Rank allocation score multiplies the rounded sample-mean retained rank by singular-value mass summed over all n_s samples:

$$s_{l,j,b} = \text{round}\left(\frac{1}{n_s} \sum_{q=1}^{n_s} r_\tau(\mathbf{E}_{l,j,b}^{(q)}) \right) \sum_{q=1}^{n_s} \sum_i |\sigma_i(\mathbf{E}_{l,j,b}^{(q)})|^p. \tag{7}$$

The score guides allocation rather than defining a reconstruction loss. Retained rank describes the error matrix, not the adapter rank assigned under the parameter budget; larger retained rank and singular-value mass increase the allocation score.

The reference parameter budget sets total reconstruction capacity. After reserving minimum rank per precision, complementary cross-block and within-block paths distribute the remaining budget. Each linear layer allocates its assigned capacity among candidate precisions by allocation score (Appendix G.1).

3.3 PRISMLORA: SHARED AND PRECISION-SPECIFIC LOW-RANK RECONSTRUCTION

Each target linear layer stores low-rank factors $\mathbf{U} \in \mathbb{R}^{d_{\text{out}} \times R}$ and $\mathbf{V} \in \mathbb{R}^{R \times d_{\text{in}}}$. Candidate b selects $I_b = S_b \cup D_b$, where the shared index sets use prefix-based sharing, with $S_8 \subseteq S_7 \subseteq S_6 \subseteq S_5 \subseteq S_4$, and the precision-specific sets D_b are disjoint. Lower precisions can therefore reuse shared low-rank components while retaining precision-specific components. The complete sets I_b need not be nested.

The low-rank adapter for each precision forms the correction

$$\Delta \mathbf{W}_b = \mathbf{U}_{:,I_b} \mathbf{V}_{I_b,:}. \tag{8}$$

Storage covers the longest shared prefix and precision-specific segments. $\mathbf{U}$ is initialized to zero and $\mathbf{V}$ with Xavier initialization. The correction is added to the frozen base weight before quantization.

Weights and activations use asymmetric affine quantization. Calibration uses fake quantization with straight-through rounding. During joint block reconstruction, the quantized output is matched directly to the full-precision block output, allowing PrismLoRA to correct the combined quantization error without explicitly decomposing individual noise sources.

For transformer block F_l, PrismQ jointly optimizes clipping logits and low-rank factors against the full-precision block output while keeping the base weights frozen. The LLM reconstruction objective is

$$\min_{\Theta_l} \sum_{b \in \mathcal{B}} \mathbb{E}_{\mathbf{X} \sim \mathcal{D}_{\text{cal}}} \left\| F_l(\mathbf{X}) - \widetilde{F}_{l,b}(\mathbf{X}; \Theta_l) \right\|_1, \tag{9}$$

where Θ_l contains the clipping and low-rank parameters of block l. These losses train the block, whereas Equation 7 determines reconstruction-capacity allocation.

3.4 ST: FIXED-BUDGET CALIBRATION-STATE PROPAGATION

For sample q, let $\mathbf{Z}^{\text{fp},(q)}$ and $\mathbf{Z}_b^{(q)}$ denote the full-precision and candidate-specific quantized outputs of F_l and $\widetilde{F}_{l,b}$, respectively. Let $\mathbf{Z}_t^{\text{fp},(q)}$ and $\mathbf{Z}_{b,t}^{(q)}$ denote their token-t hidden states. For candidate b and hidden width h, ST measures the LLM token-wise quantization error as

$$e_{b,t}^{(q)} = \frac{1}{h} \left\| \mathbf{Z}_{b,t}^{(q)} - \mathbf{Z}_t^{\text{fp},(q)} \right\|_1. \tag{10}$$

ST sorts token errors within each candidate and normalizes them by the candidate total. It averages the resulting marginal utilities across the batch. Water filling assigns the next unit of cache budget to the largest remaining normalized marginal, subject to

$$\sum_{b\in\mathcal{B}} K_b = K, \qquad \lceil n\rho_{\min}\rceil \le K_b \le \lfloor n\rho_{\max}\rfloor. \tag{11}$$

Here, K is the total cache budget, K_b is the budget assigned to candidate b, and $\rho_{\min}$ and $\rho_{\max}$ bound the fraction of token states assigned to each candidate.

For each sample, let T_b contain the indices of the K_b token states with the largest token-wise quantization errors for candidate b. Omitting the sample index, ST replaces full-precision states at these positions with candidate-specific quantized states:

$$\widehat{\mathbf{Z}}_{b,t} = \begin{cases} \mathbf{Z}_{b,t}, & t \in T_b, \\ \mathbf{Z}_t^{\mathrm{fp}}, & t \notin T_b. \end{cases} \tag{12}$$

These calibration states carry candidate-specific quantization errors into the calibration inputs of the next transformer block. Unselected positions remain full precision. ST operates during calibration-state propagation and does not change token processing in the deployed forward pass.

4 Experiments

We evaluate LLaMA-2-7B/13B (Touvron et al., 2023), LLaMA-3-8B (Grattafiori et al., 2024), and Qwen2-7B-Instruct (Yang et al., 2024). PrismQ LLM calibration uses 128 WikiText-2 sequences (Merity et al., 2017) of 2048 tokens with asymmetric weights and activations. VLM runs also use asymmetric activations, with MSE reconstruction.

We report WikiText-2/C4 (Raffel et al., 2020) PPL and mean zero-shot accuracy on PIQA, ARC-Easy/Challenge, HellaSwag, and WinoGrande (Bisk et al., 2020; Clark et al., 2018; Zellers et al., 2019; Sakaguchi et al., 2020) using lm-evaluation-harness (Gao et al., 2023). Local means use normalized accuracy except for WinoGrande. Runs use one RTX 6000 Ada; Appendix G gives the complete settings and timing details.

4.1 Main results

LLM results. Table 1 compares full-precision and quantized results, including GPTAQ (Li et al., 2025) and QLLM (Liu et al., 2024); R, Λ, and C denote rotation, diagonal scaling, and channel reassembly. PrismQ exceeds reproduced QuEPT Avg. in 18 of 20 model–precision pairs, with a mean gain of 3.48 percentage points. QuEPT leads at LLaMA-3-8B W8A8 and Qwen2-7B W4A4 by 0.22 and 0.26 points, respectively. PrismQ lowers WikiText-2 PPL in every pair and matches or lowers C4 PPL. Appendices A and B give task scores; Appendix A.1 reports KV4 results. Appendix G details baseline transformations and averaging.

Table 2 compares calibration cost. Single-target methods are projected over a Cartesian grid of N weight and M activation targets, whereas QuEPT and PrismQ jointly calibrate five formats. GPTAQ uses model-specific single-target means, with timing intervals given in Appendix G. PrismQ requires less joint calibration time than the QuEPT reproduction across all four evaluated models.

Figure 3 evaluates LLaMA-2-7B with FP16 activations. Uniform PrismQ improves WikiText-2 PPL over QuEPT at 2 and 3 bits, while QuEPT is slightly better at 4 and 6 bits. Near 2.25 average bits, mixed PrismQ reaches 8.84 PPL versus QuEPT's 8.97, but mixed precision does not consistently improve over uniform quantization. Appendix C reports all results, sources, allocation settings, and sensitivity-analysis overhead.

ViT results. Table 3 reports ImageNet-1K top-1 accuracy and same-device calibration time. FlexiQ (Kim et al., 2026) uses average weight/activation budgets at 5–7 bits rather than uniform formats. PrismQ matches or improves the best displayed accuracy at 18 of 20 operating points with short joint calibration times. Appendix G gives sources and settings.

Table 1: Language-model PPL (↓) and five-task Avg. (%, ↑). FP is FP16 except for Qwen2-7B-Instruct (BF16). OFA: ✓, joint calibration; ✗, format-specific calibration. Gray marks PrismQ; bold marks the best quantized result per model and precision, including ties. Dashes denote unreported results.

Bits	Method	Trans.	OFA	LLaMA-2-7B			LLaMA-2-13B			LLaMA-3-8B			Qwen2-7B		
				Wiki2	C4	Avg.	Wiki2	C4	Avg.	Wiki2	C4	Avg.	Wiki2	C4	Avg.
FP	-	-	-	5.47	6.97	69.05	4.88	6.47	71.72	6.14	8.89	72.86	7.60	10.24	72.29
W4A4	OmniQuant	Λ	✗	14.26	18.02	49.13	12.30	14.55	53.13	-	-	-	-	-	-
	QLLM	C	✗	11.75	13.26	51.60	9.09	11.13	54.31	-	-	-	-	-	-
	QuaRot	R	✗	7.82	9.65	59.73	6.18	8.02	66.32	10.93	14.52	59.00	9.39	12.52	66.58
	DuQuant	R, Λ	✗	6.28	7.90	59.11	5.42	7.05	62.84	8.56	11.98	65.05	-	-	-
	GPTAQ	R	✗	5.86	7.72	65.03	**5.17**	7.00	67.15	7.39	11.67	67.94	-	-	-
	ABQ-LLM	Λ	✗	9.31	12.85	50.62	8.62	11.47	53.33	-	-	-	-	-	-
	QuEPT	R	✓	5.99	7.72	59.08	5.30	7.03	62.94	7.64	11.39	67.03	8.66	12.00	**69.70**
	PrismQ	R	✓	**5.85**	**7.55**	**65.98**	**5.17**	**6.88**	**70.02**	**7.35**	**10.81**	**69.32**	**8.27**	**11.36**	69.43
W5A5	QuaRot	R	✗	6.53	8.24	64.58	5.48	7.14	69.93	7.82	11.14	68.14	8.27	11.11	70.15
	GPTAQ	R	✗	**5.57**	7.16	66.30	**4.96**	6.61	68.97	**6.44**	9.56	70.93	-	-	-
	QuEPT	R	✓	5.62	7.16	61.25	5.01	6.61	64.66	6.61	9.59	71.48	7.84	10.61	71.47
	PrismQ	R	✓	**5.57**	**7.12**	**67.80**	4.97	**6.57**	**71.62**	6.53	**9.44**	**72.16**	**7.80**	**10.56**	**72.09**
W6A6	OmniQuant	Λ	✗	5.87	7.48	58.56	5.14	6.74	64.71	7.24	9.82	69.51	-	-	-
	SmoothQuant	Λ	✗	6.20	7.76	61.40	5.18	6.76	64.30	7.07	9.57	70.50	-	-	-
	QLLM	C	✗	5.91	7.31	61.43	5.08	6.71	64.87	-	-	-	-	-	-
	QuaRot	R	✗	5.93	7.51	67.59	5.19	6.81	70.44	7.04	10.15	70.17	8.00	10.77	70.67
	DuQuant	R, Λ	✗	5.53	7.03	62.13	4.92	**6.50**	64.88	6.27	**8.38**	72.29	-	-	-
	GPTAQ	R	✗	5.51	7.03	66.23	**4.91**	6.52	69.24	**6.22**	9.09	71.85	-	-	-
	ABQ-LLM	Λ	✗	5.63	7.21	61.69	5.00	6.64	64.38	-	-	-	-	-	-
	QuEPT	R	✓	5.52	7.03	61.88	4.92	6.51	65.59	6.30	9.13	72.57	**7.65**	10.36	71.70
	PrismQ	R	✓	**5.50**	**7.01**	**68.79**	**4.91**	**6.50**	**71.32**	6.26	9.05	**72.64**	**7.65**	**10.33**	**72.18**
W7A7	QuaRot	R	✗	5.55	7.07	68.41	4.94	6.53	71.50	6.35	9.15	72.63	7.67	10.36	72.09
	QuEPT	R	✓	5.50	6.99	62.22	4.90	**6.48**	65.53	6.19	8.96	72.66	7.64	**10.27**	72.01
	PrismQ	R	✓	**5.48**	**6.98**	**68.73**	**4.89**	**6.48**	**71.58**	**6.18**	**8.95**	**72.78**	**7.61**	**10.27**	**72.17**
W8A8	QuaRot	R	✗	5.50	6.99	68.82	4.90	6.48	**71.80**	6.20	8.96	72.59	7.63	10.27	72.20
	GPTAQ	R	✗	5.49	**6.98**	66.52	**4.89**	6.49	69.35	**6.16**	8.96	71.87	-	-	-
	ABQ-LLM	Λ	✗	5.48	6.99	62.08	**4.89**	**6.47**	65.35	-	-	-	-	-	-
	QuEPT	R	✓	5.49	**6.98**	62.44	4.90	**6.47**	65.49	6.17	8.94	**72.78**	7.62	10.26	72.11
	PrismQ	R	✓	**5.47**	**6.98**	**68.93**	**4.89**	**6.47**	71.74	**6.16**	**8.92**	72.56	**7.60**	**10.25**	**72.23**

Table 2: Calibration time (hours). Single-target costs are projected as $t \times N \times M$; QuEPT and PrismQ report joint time over the five evaluated formats. A dash denotes unreported calibration time.

Method	Once-for-All	LLaMA-2-7B	LLaMA-2-13B	LLaMA-3-8B	Qwen2-7B
OmniQuant	✗	$3.21 \times N \times M$	$5.10 \times N \times M$	-	-
QLLM	✗	$1.79 \times N \times M$	$2.73 \times N \times M$	-	-
GPTAQ	✗	$0.22 \times N \times M$	$0.53 \times N \times M$	$0.27 \times N \times M$	-
QuEPT	✓	4.40	8.14	4.81	4.52
PrismQ	✓	3.27	6.19	3.60	3.41

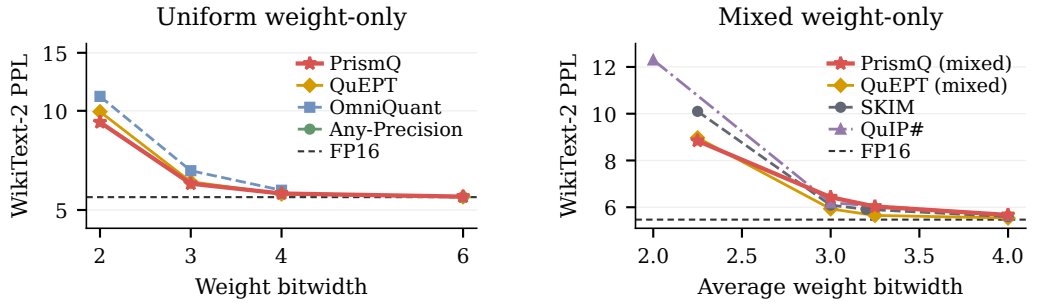


Figure 3: LLaMA-2-7B weight-only WikiText-2 PPL with 16-bit activations. Left: uniform weight precision. Right: mixed average weight precision. The FP16 reference is 5.47; lines guide the eye.

VLM results. Table 4 evaluates LLaVA-OneVision-Qwen2-7B with quantization applied only to the language decoder. Baselines and FP16 follow QuEPT Table 3 (Xu et al., 2026b). Avg. is calculated

Table 3: ImageNet-1K top-1 accuracy (%) and calibration time (hours). OFA follows Table 1. ERQ time is projected from a single-target mean; PTMQ, QuEPT, and PrismQ report joint time. Gray marks PrismQ; bold marks the best displayed accuracy. Dashes denote unreported entries.

Model	Method	OFA	W4A4	W5A5	W6A6	W7A7	W8A8	FP32	Time (h)
ViT-S	ERQ (Zhong et al., 2025a)	✗	71.6	79.1	80.7	80.9	81.2	81.3	$0.14 \times N \times M$
	PTMQ (Xu et al., 2024)	✓	-	-	76.1	77.1	78.2		4.66
	QuEPT (Xu et al., 2026b)	✓	75.1	**79.7**	80.6	81.0	81.2		0.15
	FlexiQ† (Kim et al., 2026)	✓	**77.1**	78.8	80.1	80.6	81.1		-
	PrismQ (Ours)	✓	76.2	**79.7**	**81.0**	**81.2**	**81.3**		0.14
ViT-B	ERQ (Zhong et al., 2025a)	✗	78.7	83.0	**84.0**	84.3	84.4	84.5	$0.25 \times N \times M$
	PTMQ (Xu et al., 2024)	✓	-	-	77.7	78.6	79.1		9.97
	QuEPT (Xu et al., 2026b)	✓	80.7	83.3	83.8	84.2	84.3		0.30
	FlexiQ† (Kim et al., 2026)	✓	80.2	81.4	81.7	81.9	81.4		-
	PrismQ (Ours)	✓	**81.4**	**83.4**	**84.0**	**84.4**	**84.5**		0.27
DeiT-S	ERQ (Zhong et al., 2025a)	✗	74.4	78.0	79.1	79.6	**79.7**	79.8	$0.14 \times N \times M$
	PTMQ (Xu et al., 2024)	✓	-	-	78.7	79.3	79.5		4.80
	QuEPT (Xu et al., 2026b)	✓	75.3	78.1	**79.3**	**79.7**	**79.7**		0.15
	FlexiQ† (Kim et al., 2026)	✓	**76.5**	78.0	78.4	78.6	78.7		-
	PrismQ (Ours)	✓	75.7	**78.4**	**79.3**	**79.7**	**79.7**		0.14
Swin-S	ERQ (Zhong et al., 2025a)	✗	81.2	82.5	82.9	83.1	**83.2**	83.2	$0.21 \times N \times M$
	QuEPT (Xu et al., 2026b)	✓	81.9	82.8	82.9	83.1	**83.2**		0.42
	FlexiQ† (Kim et al., 2026)	✓	81.8	82.7	82.9	83.1	83.1		-
	PrismQ (Ours)	✓	**82.9**	**83.1**	**83.2**	**83.2**	**83.2**		0.22

from unrounded local scores or published reference scores, then rounded to one decimal. PrismQ gives the highest displayed Avg. at W4A8 and remains close to QuEPT at W4A4. Appendix G provides the corresponding calibration settings and implementation details.

Table 4: Vision-language scores (0–100). OFA follows Table 1. Gray marks PrismQ; bold marks the best quantized score per precision and metric.

Bit	Method	OFA	MMMU	OCRBench	TextVQA	VizWiz	SEEDBench	Avg
FP16	-	-	48.0	62.2	76.1	60.4	72.6	63.9
W4A4	QuEPT	✓	33.9	49.1	**64.4**	**59.8**	**66.5**	**54.7**
	PrismQ	✓	**34.1**	**51.5**	63.6	57.3	66.1	54.5
W4A8	SmoothQuant	✗	30.9	32.0	56.9	56.7	41.6	43.6
	MBQ	✗	42.6	52.3	68.3	58.9	64.4	57.3
	QuEPT	✓	43.4	61.2	71.5	**61.3**	70.7	61.6
	PrismQ	✓	**45.4**	**63.9**	**73.8**	59.1	**72.2**	**62.9**

4.2 Ablation studies

Tables 5–8 evaluate clipping-induced-error analysis, allocation scores, shared and precision-specific low-rank structures, and calibration-state propagation on LLaMA-2-7B. All runs use asymmetric WikiText-2 calibration and report WikiText-2/C4 PPL at W4A4, W6A6, and W8A8. Changing the allocation score leaves the MAE reconstruction objective unchanged. Appendix E defines the switches, parameter budgets, and propagation variants.

Using the clipping-induced error together with precision-specific rank reduces W4A4 PPL from 5.96/7.64 to 5.85/7.55. Power×Rank gives the lowest W4A4 PPL among the allocation scores, while differences become small at higher precision. Under the same rank-40 reference parameter budget, PrismQ improves W4A4 PPL over fully shared, cascade-only, and fully precision-specific reconstruction. ST also gives the lowest W4A4 PPL among the calibration-state propagation strategies by selecting quantized token states according to token-wise quantization error. These gains

Clipping	Prec.	W4A4	W6A6	W8A8
✗	✗	5.96/7.64	5.52/7.03	**5.47/6.98**
✗	✓	5.90/7.61	5.51/**7.01**	**5.47/6.98**
✓	✗	5.95/7.64	5.51/**7.01**	**5.47/6.98**
✓	✓	**5.85/7.55**	**5.50/7.01**	**5.47/6.98**

Table 5: Clipping-induced-error ablation. Prec. denotes precision-specific rank.

Score	W4A4	W6A6	W8A8
MAE	5.97/7.63	5.51/7.02	**5.47/6.98**
MSE	5.96/7.64	5.52/7.03	**5.47/6.98**
QSNR	5.98/7.65	5.51/**7.01**	**5.47/6.98**
Power×Rank	**5.85/7.55**	**5.50/7.01**	**5.47/6.98**

Table 6: Allocation-score ablation. All variants use MAE reconstruction.

Structure	W4A4	W6A6	W8A8
Shared	6.00/7.78	5.53/7.03	5.48/6.99
Cascade	5.95/7.62	**5.50/7.01**	**5.47/6.98**
Precision-specific	5.96/7.63	**5.50/7.01**	**5.47/6.98**
PrismQ	**5.85/7.55**	**5.50/7.01**	**5.47/6.98**

Table 7: Low-rank sharing ablation. All variants use the same parameter budget.

Strategy	W4A4	W6A6	W8A8
Full precision	6.01/7.67	5.52/7.02	**5.47/6.98**
Fusion Token	5.99/7.66	5.52/7.02	5.48/**6.98**
Random Token	5.98/7.65	5.52/7.02	5.48/**6.98**
ST (default)	**5.85/7.55**	**5.50/7.01**	**5.47/6.98**

Table 8: Calibration-state ablation. Quantized-state variants use matched cache budgets.

concentrate at W4A4; several W8A8 configurations tie at the reported precision, so the ablations provide clearer separation between designs at low bit widths. Appendix E gives configurations and rank statistics; Appendix E.2 compares cache budgets.

4.3 Deployment memory and latency

Figure 4 shows deployment measurements for LLaMA-2-7B with the ABQ-LLM kernel framework. At prompt length 512, W4A4 reduces sampled peak memory and achieves $1.650\times$ E2E speedup over FP16; W8A8 reduces memory but remains slower. Appendix D reports both prompt lengths and sampling details, while Appendix G specifies timing boundaries.

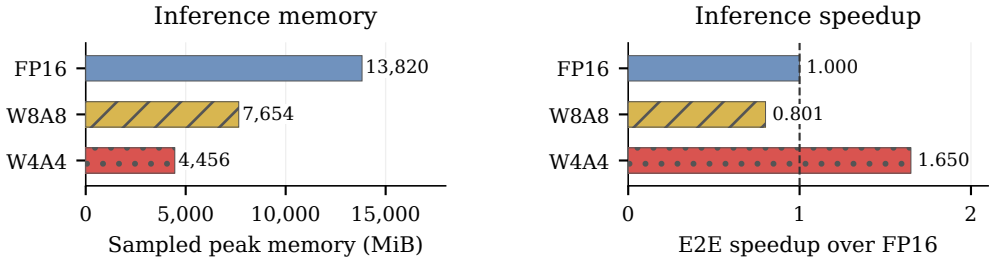


Figure 4: LLaMA-2-7B deployment on one RTX 6000 Ada GPU with batch size one, a 512-token prompt, and 128 generated tokens. Left: sampled peak process memory over warmed inference runs. Right: E2E speedup over FP16; the dashed line marks $1\times$.

5 Conclusion

PrismQ introduces noise-aware resource allocation for elastic post-training quantization. PrismRank uses clipping-induced error to allocate reconstruction capacity across target linear layers and candidate precisions, while PrismLoRA realizes shared and precision-specific low-rank reconstruction and ST performs fixed-budget calibration-state propagation. LLM and ImageNet experiments demonstrate consistent performance across multiple operating points, with reconstruction ranks that vary across precision and linear layer. The experiments assess reconstruction quality alongside joint calibration cost and deployment performance, with measured memory savings that do not always translate into lower inference latency. The current theoretical formulation and method have not yet been extended to FP quantization formats, leaving this extension for future work.

AI USE STATEMENT

Generative AI tools assist English translation, organization, literature discovery, code implementation, debugging, and result organization. The authors verify the bibliography, claims, code, and final manuscript, take responsibility for all artifacts, and do not use generative AI to run experiments.

ETHICS STATEMENT

This work studies post-training quantization methods for neural networks and uses publicly available datasets and benchmarks for evaluation. It does not involve human participants or personally identifiable information. We do not identify any direct ethical concerns arising from the proposed methodology or experimental evaluation. Nevertheless, practical deployment of quantized models in safety-critical applications should include appropriate validation and human oversight under the target operating conditions.

REPRODUCIBILITY STATEMENT

We provide the model architecture, quantization settings, training objectives, implementation details, datasets, and evaluation metrics in the main paper. Additional implementation details, experimental settings, and ablation results are included in the appendix and supplementary material.

REFERENCES

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

Runsheng Bai, Bo Liu, and Qiang Liu. SKIM: Any-bit quantization pushing the limits of post-training quantization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 2520–2534, 2025.

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

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439, 2020.

Yelysei Bondarenko, Riccardo Del Chiaro, and Markus Nagel. Low-rank quantization-aware training for LLMs. *arXiv preprint arXiv:2406.06385*, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Øyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations*, 2023.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 2023.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Han Guo, Philip Greengard, Eric P. Xing, and Yoon Kim. LQ-LoRA: Low-rank plus quantized matrix decomposition for efficient language model finetuning. In *International Conference on Learning Representations*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Jaemin Kim, Hongjun Um, Sungkyun Kim, Yongjun Park, and Jiwon Seo. FlexiQ: Adaptive mixed-precision quantization for latency/accuracy trade-offs in deep neural networks. In *Proceedings of the 2026 European Conference on Computer Systems*, pp. 1298–1313. Association for Computing Machinery, 2026.

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

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. LoftQ: LoRA-fine-tuning-aware quantization for large language models. In *International Conference on Learning Representations*, 2024.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. GPTAQ: Efficient finetuning-free quantization for asymmetric calibration. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36690–36706, 2025.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. DuQuant: Distributing outliers via dual transformation makes stronger quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 87766–87800, 2024a.

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. In *Proceedings of Machine Learning and Systems*, volume 6, pp. 87–100, 2024b.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision – ECCV 2014*, pp. 740–755, 2014.

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

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Pranav Ajit Nair, Puranjay Datta, Jeff Dean, Prateek Jain, and Aditya Kusupati. Matryoshka quantization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 45484–45506, 2025.

Yeonhong Park, Jake Hyun, Sanglyul Cho, Bonggeun Sim, and Jae W. Lee. Any-precision LLM: Low-cost deployment of multiple, different-sized LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39682–39701, 2024.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, 2020.

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 *International Conference on Learning Representations*, 2024.

Moran Shkolnik, Brian Chmiel, Ron Banner, Gil Shomron, Yury Nahshan, Alex Bronstein, and Uri Weiser. Robust quantization: One model to rule them all. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5308–5317, 2020.

Hugo Touvron et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

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*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48630–48656, 2024.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099, 2023.

Ke Xu, Lei Han, Ye Tian, Shangshang Yang, and Xingyi Zhang. EQ-Net: Elastic quantization neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1505–1514, 2023.

Ke Xu, Zhongcheng Li, Shanshan Wang, and Xingyi Zhang. PTMQ: Post-training multi-bit quantization of neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(14): 16193–16201, 2024.

Ke Xu, Tianyu Ma, Xiaozheng Shen, Zining Wang, Xiaoyun Wang, and Jinyang Guo. PEQuant: Prompt-driven activation reconstruction for elastic quantization of vision transformers. In *Proceedings of the 34th ACM International Conference on Multimedia*, 2026a.

Ke Xu, Yixin Wang, Zhongcheng Li, Hao Cui, Jinshui Hu, and Xingyi Zhang. QuEPT: Quantized elastic precision transformers with one-shot calibration for multi-bit switching. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(32):27287–27295, 2026b.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024.

Rongguang Ye, Ming Tang, and Edith C. H. Ngai. On-the-fly adaptation to quantization: Configuration-aware LoRA for efficient fine-tuning of quantized LLMs. In *International Conference on Learning Representations*, 2026.

Haichao Yu, Haoxiang Li, Humphrey Shi, Thomas S. Huang, and Gang Hua. Any-precision deep neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10763–10771, 2021.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Chao Zeng, Songwei Liu, Yusheng Xie, Hong Liu, Xiaojian Wang, Miao Wei, Shu Yang, Fangmin Chen, and Xing Mei. ABQ-LLM: Arbitrary-bit quantized inference acceleration for large language models. *arXiv preprint arXiv:2408.08554*, 2024.

Jiaqi Zhao, Miao Zhang, Chao Zeng, Ming Wang, Xuebo Liu, and Liqiang Nie. LRQuant: Learnable and robust post-training quantization for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 2240–2255, 2024.

Jiaqi Zhao, Chao Zeng, Ming Wang, Linxuan Han, Yuzhang Shang, Miao Zhang, and Liqiang Nie. LRQuant+: A unified and learnable framework to post-training quantization for transformer-based large foundation models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(12):11817–11833, 2025.

Yunshan Zhong, You Huang, Jiawei Hu, Yuxin Zhang, and Rongrong Ji. Towards accurate post-training quantization of vision transformers via error reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):2676–2692, 2025a.

Yunshan Zhong, Yuyao Zhou, Fei Chao, and Rongrong Ji. Mbquant: A novel multi-branch topology method for arbitrary bit-width network quantization. *Pattern Recognition*, 158:111061, 2025b.

A Full Language-Model Results

Tables 9, 10, and 11 expand the local results into task-level metrics. The five-task mean is computed from unrounded values in the evaluation logs and is shown to two decimals.

Table 9: PrismQ LLaMA-2 results with calibration seed 42. PPL is WikiText-2/C4, and accuracy values are percentages.

Model	Bit	PPL	PIQA	ARC-E	ARC-C	HellaSwag	WinoGrande	Avg
7B	W4A4	5.85/7.55	76.99	71.17	42.41	73.28	66.06	65.98
7B	W5A5	5.57/7.12	77.97	73.44	43.60	75.56	68.43	67.80
7B	W6A6	5.50/7.01	78.67	74.07	45.99	75.60	69.61	68.79
7B	W7A7	5.48/6.98	79.16	74.12	45.56	75.82	68.98	68.73
7B	W8A8	5.47/6.98	79.16	74.66	45.90	76.00	68.90	68.93
13B	W4A4	5.17/6.88	79.27	75.55	47.18	77.00	71.11	70.02
13B	W5A5	4.97/6.57	80.20	77.02	48.81	78.77	73.32	71.62
13B	W6A6	4.91/6.50	80.41	77.31	47.95	79.19	71.74	71.32
13B	W7A7	4.89/6.48	80.47	77.44	48.72	79.39	71.90	71.58
13B	W8A8	4.89/6.47	80.41	77.65	49.23	79.44	71.98	71.74

Table 10: PrismQ LLaMA-3-8B results with calibration seed 42 and ST. PPL is WikiText-2/C4, and accuracy values are percentages.

Bit	PPL	PIQA	ARC-E	ARC-C	HellaSwag	WinoGrande	Avg
W4A4	7.35/10.81	78.35	74.12	47.27	75.53	71.35	69.32
W5A5	6.53/9.44	80.25	77.53	51.62	78.15	73.24	72.16
W6A6	6.26/9.05	80.36	77.06	53.41	78.79	73.56	72.64
W7A7	6.18/8.95	80.52	77.90	53.41	78.99	73.09	72.78
W8A8	6.16/8.92	80.74	77.78	52.56	79.20	72.53	72.56

Table 11: Qwen2-7B-Instruct results from completed seed-42 local evaluations. The full-precision reference uses BF16. PPL is WikiText-2/C4; task accuracies and their mean are percentages.

Method	Bit	PPL	PIQA	ARC-E	ARC-C	HellaSwag	WinoGrande	Avg
Full precision	BF16	7.60/10.24	80.58	76.64	54.10	80.67	69.46	72.29
QuaRot	W4A4	9.39/12.52	76.88	70.16	47.35	75.45	63.06	66.58
	W5A5	8.27/11.11	79.92	73.99	51.96	78.32	66.54	70.15
	W6A6	8.00/10.77	79.33	75.72	53.67	78.92	65.75	70.67
	W7A7	7.67/10.36	80.63	77.10	54.10	80.21	68.43	72.09
	W8A8	7.63/10.27	80.09	76.60	53.84	80.47	70.01	72.20
PrismQ	W4A4	8.27/11.36	79.00	73.78	51.62	77.57	65.19	69.43
	W5A5	7.80/10.56	80.20	76.35	53.41	80.58	69.93	72.09
	W6A6	7.65/10.33	80.03	76.39	54.69	79.87	69.93	72.18
	W7A7	7.61/10.27	80.47	75.80	53.50	80.50	70.56	72.17
	W8A8	7.60/10.25	80.41	76.35	54.18	80.65	69.53	72.23

A.1 KV-cache quantization

Table 12 reports a separate LLaMA-2-7B run with 4-bit keys and values at every evaluated weight–activation format. Calibration and evaluation follow the main LLM settings in Appendix G, with two KV-specific changes. Keys and values use asymmetric affine fake quantization in groups of 128 elements, with a clipping ratio of 0.95. Online Hadamard transforms for queries and keys are enabled: keys are quantized after RoPE and the transform, while values are quantized at the value-projection output. The main LLM results retain 16-bit keys and values.

Table 12: LLaMA-2-7B with K4V4. PPL is WikiText-2/C4; task scores and Avg. are percentages. PIQA, ARC-E, ARC-C, and HellaSwag use normalized accuracy; WinoGrande uses accuracy. Avg. is computed from unrounded task scores.

Format	PPL	PIQA	ARC-E	ARC-C	HellaSwag	WinoGrande	Avg.
W4A4	5.90/7.60	76.17	71.00	42.41	70.65	66.54	65.35
W5A5	5.60/7.16	77.31	72.85	44.11	72.82	67.25	66.87
W6A6	5.53/7.05	77.53	73.19	45.56	73.43	67.88	67.52
W7A7	5.51/7.02	77.42	73.91	44.88	73.33	68.67	67.64
W8A8	5.50/7.01	77.64	73.48	45.14	73.33	69.14	67.75

B QuEPT Reproduction

Table 13 expands the QuEPT results in Table 1. Each model uses one joint calibration on WikiText-2 with seed 2 and 15 epochs. PIQA, ARC-E, ARC-C, and HellaSwag use normalized accuracy; WinoGrande uses accuracy. Means use the unrounded logged fractions.

Table 13: QuEPT reproduction with calibration seed 2. PPL is WikiText-2/C4; task scores and their mean are percentages.

Model	Bit	PPL	PIQA	ARC-E	ARC-C	HellaSwag	WinoGrande	Avg.
LLaMA-2 7B	W4A4	5.99/7.72	75.73	50.63	37.80	69.10	62.12	59.08
	W5A5	5.62/7.16	76.44	52.90	39.25	72.15	65.51	61.25
	W6A6	5.52/7.03	77.37	52.69	40.27	72.62	66.46	61.88
	W7A7	5.50/6.99	77.26	52.69	41.04	72.87	67.25	62.22
	W8A8	5.49/6.98	77.09	53.54	41.13	72.94	67.48	62.44
LLaMA-2 13B	W4A4	5.30/7.03	77.26	55.01	42.15	73.34	66.93	62.94
	W5A5	5.01/6.61	79.05	56.78	44.11	75.26	68.11	64.66
	W6A6	4.92/6.51	79.38	57.87	44.28	76.39	70.01	65.59
	W7A7	4.90/6.48	79.05	58.33	44.54	76.36	69.38	65.53
	W8A8	4.90/6.47	79.16	58.00	44.37	76.61	69.30	65.49
LLaMA-3 8B	W4A4	7.64/11.39	77.64	71.93	45.39	73.96	66.22	67.03
	W5A5	6.61/9.59	79.87	77.40	50.85	77.79	71.51	71.48
	W6A6	6.30/9.13	81.18	77.82	52.39	78.63	72.85	72.57
	W7A7	6.19/8.96	80.30	78.07	52.90	78.77	73.24	72.66
	W8A8	6.17/8.94	80.63	78.16	52.99	78.97	73.16	72.78
Qwen2 7B	W4A4	8.66/12.00	76.77	75.59	52.05	76.59	67.48	69.70
	W5A5	7.84/10.61	79.65	76.81	53.67	79.41	67.80	71.47
	W6A6	7.65/10.36	79.60	76.01	53.33	80.10	69.46	71.70
	W7A7	7.64/10.27	80.69	76.14	54.44	80.44	68.35	72.01
	W8A8	7.62/10.26	80.58	76.26	53.92	80.59	69.22	72.11

C Weight-Only Results

Tables 14 and 15 report LLaMA-2-7B weight-only PPL with 16-bit activations. S9 and S11 denote Tables 9 and 11 of the QuEPT supplementary material, respectively. These are published reference results, rather than same-protocol reproductions. The FP16 reference from S9 is 5.47 on WikiText-2 and 6.97 on C4. The uniform QuEPT W4 entry follows S9 (5.59); S11 reports 5.54 for that cell. Its W6 entry follows S11. Missing cells remain unreported. Uniform comparisons follow the published references in the QuEPT supplementary material; mixed comparisons also include SqueezeLLM (Kim et al., 2024), QuIP# (Tseng et al., 2024), and SKIM (Bai et al., 2025).

PrismQ uses WikiText-2 with 128 training and 64 validation sequences of length 2048, seed 42, 12 reconstruction epochs, and MAE loss. Weight candidates span 2–8 bits with FP16 activations; W2 uses groups of 128 input channels, and higher precisions use per-output-channel quantization. Hadamard pre-rotation and ST calibration-state propagation remain enabled. The reference parameter budget corresponds to rank 60 at each target linear layer, with the minimum rank per precision and precision-specific rank both set to two. This weight-only run is separate from the main weight–activation experiment. The mixed evaluation uses MAE sensitivity and greedy upgrades across candidate weight formats, whereas QuEPT uses KL sensitivity with dynamic programming. PrismQ sensitivity analysis adds 36.49 hours. Actual PrismQ bitwidths are parameter-count-weighted averages over quantized linear-layer weights and exclude quantization metadata. They can differ slightly from the target budgets; published comparison bitwidths are retained as reported.

Table 14: Uniform weight-only LLaMA-2-7B results with FP16 activations. Lower PPL is better. Source abbreviations refer to QuEPT supplementary tables; a dash denotes an unreported value.

Method	Weight bits	WikiText-2	C4	Source
GPTQ	2	36.77	33.70	S11
GPTQ	3	8.37	9.81	S11
GPTQ	4	5.83	7.37	S11
GPTQ	6	5.49	7.01	S11
AWQ	2	2.20×10^5	1.70×10^5	S11
AWQ	3	24.00	23.85	S11
AWQ	4	6.15	7.68	S11
AWQ	6	5.51	7.02	S11
OmniQuant	2	11.06	15.02	S11
OmniQuant	3	6.58	8.65	S11
OmniQuant	4	5.74	7.35	S11
SpinQuant	4	5.58	-	S11
Any-Precision	4	5.62	7.45	S11
Any-Precision	6	5.47	7.27	S11
QuEPT	2	9.94	11.05	S9
QuEPT	3	6.09	7.57	S9
QuEPT	4	5.59	7.12	S9
QuEPT	6	5.48	7.00	S11
PrismQ	2	9.23	13.34	This work
PrismQ	3	6.01	7.84	This work
PrismQ	4	5.61	7.17	This work
PrismQ	6	5.49	6.99	This work

Table 15: Mixed weight-only comparison with FP16 activations. Average bits are published values for baselines and logged values for PrismQ; target budgets are shown for PrismQ. QuEPT uniform is included as a reference. Missing values are not interpolated.

Method	Target bits	Average bits	WikiText-2	C4	Source
SqueezeLLM	-	2.23	-	-	S9
SqueezeLLM	-	3	6.18	7.72	S9
SqueezeLLM	-	3.24	5.96	7.51	S9
SqueezeLLM	-	4	5.62	7.12	S9
OmniQuant	-	2.25	11.06	15.02	S9
OmniQuant	-	3	6.58	8.65	S9
OmniQuant	-	3.25	6.03	7.75	S9
OmniQuant	-	4	5.74	7.35	S9
QuIP#	-	2	12.30	14.80	S9
QuIP#	-	3	6.19	7.85	S9
QuIP#	-	4	5.66	7.17	S9
SKIM	-	2.25	10.10	12.42	S9
SKIM	-	3	6.09	7.66	S9
SKIM	-	3.2	5.91	7.48	S9
SKIM	-	4	5.60	7.11	S9
QuEPT (uniform)	-	2	9.94	11.05	S9
QuEPT (uniform)	-	3	6.09	7.57	S9
QuEPT (uniform)	-	4	5.59	7.12	S9
QuEPT (mixed)	-	2.25	8.97	9.07	S9
QuEPT (mixed)	-	3	5.93	7.37	S9
QuEPT (mixed)	-	3.25	5.65	7.17	S9
QuEPT (mixed)	-	4	5.54	7.09	S9
PrismQ (mixed)	2.25	2.2490	8.84	12.44	This work
PrismQ (mixed)	3.0	2.9997	6.43	8.48	This work
PrismQ (mixed)	3.25	3.2495	6.03	7.82	This work
PrismQ (mixed)	4.0	3.9995	5.67	7.25	This work

D Deployment Measurements

Table 16 provides the complete measurements underlying Figure 4, including resident memory, timing variability, and decode throughput. The plotted speedups use the ratio of the reported E2E means; no uncertainty estimate is inferred from these summary statistics.

Table 16: LLaMA-2-7B deployment on one RTX 6000 Ada GPU, with batch size one and 128 generated tokens. Resident is NVML process memory after loading; sampled peak is the maximum over five warmed inference runs at a target 10 ms sampling interval. E2E is mean $\pm$ sample standard deviation over 20 separate timing runs; both measurements use three warmups. Decode reports mean throughput. Speedup is the ratio of FP16 to format E2E means at the same prompt length.

Format	Prompt (tokens)	Resident (MiB) $\downarrow$	Sampled peak (MiB) $\downarrow$	E2E (ms) $\downarrow$ mean $\pm$ std	Decode (tok/s) $\uparrow$	Speedup $\times$ FP16 $\uparrow$
FP16	128	13322	13532	2123.200 ± 18.503	60.435	1.000
	512	13322	13820	2226.706 ± 21.187	58.491	1.000
W8A8	128	7124	7334	2395.727 ± 20.251	55.041	0.886
	512	7124	7654	2778.982 ± 32.182	53.418	0.801
W4A4	128	3926	4136	1199.890 ± 1.184	108.481	1.769
	512	3926	4456	1349.787 ± 19.212	102.276	1.650

E Allocation and Calibration-State Propagation Ablations

E.1 Ablation configurations

The four studies in Tables 5–8 share the LLaMA-2-7B default from Table 1. Calibration uses seed 42, WikiText-2 with 128 training and 64 validation sequences of 2048 tokens, asymmetric weight and activation quantization, 12 reconstruction epochs, and MAE loss. Each variant evaluates only W4A4, W6A6, and W8A8 PPL; zero-shot accuracy is not evaluated. W4 uses input-channel groups of 128, while W6 and W8 use per-output-channel quantization.

The clipping-induced-error ablation toggles rounding in the error used for reconstruction-capacity allocation and sets precision-specific rank to zero or two, with minimum rank per precision two. The score ablation replaces Power×Rank with MAE, MSE, or QSNR while retaining clipping-induced-error analysis and precision-specific rank two. These scores allocate reconstruction capacity; they do not replace the reconstruction objective.

Table 7 compares Shared, Cascade, Precision-specific, and PrismQ. Shared applies one shared rank-40 low-rank adapter to every precision. Cascade uses prefix-based sharing with ranks 8, 16, 24, 32, and 40 for W8A8 through W4A4, without precision-specific low-rank components. Precision-specific assigns a disjoint rank-8 adapter to each of the five precisions. For Cascade and Precision-specific, the minimum rank reservation uses the entire rank-40 reference parameter budget: $5 \times 8 = 40$. Although their runs enable the PrismRank analysis procedure, no budget remains for additional noise-aware allocation of reconstruction capacity. PrismQ instead reserves precision-specific rank two per candidate and allocates the remaining reconstruction capacity through the $(0, 0.5, 0.5)$ mixture. The three fixed structures have equal total factor storage per linear layer, while their per-precision ranks differ; PrismQ distributes its reference parameter budget across linear layers and candidate precisions.

Full-precision propagation forwards the reference block output. Fusion Token forwards the arithmetic mean of all candidate quantized block outputs. Random Token samples candidate-specific quantized token states using seed 42 and the same total cache budget as ST. ST determines the candidate cache budgets from token-wise quantization error and selects the token positions with the largest error. Calibration-state propagation changes calibration inputs only; it adds no inference-time token selection.

E.2 Calibration-state cache budget

Table 17 extends Table 8 with different ST cache budgets. Let r denote `st_ratio`, the total quantized-state budget relative to one complete block-output tensor over the calibration set. The $r = 0$ column reuses the Full precision propagation reference, which retains no quantized states; it is not a separate ST run with the default positive minimum quotas. Fusion Token, Random Token, and $r = 1$ reuse the existing results, with $r = 1$ matching the LLaMA-2-7B main experiment. The $r = 2$ and $r = 4$ runs use the same general calibration settings and evaluate only PPL.

For 128 sequences of 2048 tokens, hidden width 4096, and FP16 storage, one complete decoder-block output occupies $128 \times 2048 \times 4096 \times 2/2^{30} = 2$ GiB. The total feature storage includes the full-precision reference and retained quantized calibration states: Full precision uses one copy, Fusion Token and Random Token use two copies, and ST uses $1 + r$ copies. The theoretical storage estimation follows the above formulation, and the measured feature storage is consistent with the theoretical results. These measurements report the actual feature storage cost and exclude metadata, temporary tensors, model weights, and other memory overheads.

The $r = 4$ run raises `stplus_max_ratio` from 0.5 to 1.0. With five candidate formats, the default per-candidate upper bound permits a total ratio of at most $5 \times 0.5 = 2.5$. Thus $r = 4$ changes both the total cache budget and the per-candidate upper bound; the remaining ST settings follow the default.

At $r = 1$, ST improves W4A4 PPL over Fusion Token and Random Token with the same total feature storage. Increasing the budget gives no consistent improvement: $r = 2$ lowers W4A4 C4 PPL from 7.55 to 7.54 but raises WikiText-2 PPL from 5.85 to 5.86, while $r = 4$ gives 5.87/7.55. W6A6 and W8A8 results remain unchanged across $r = 1, 2, 4$ at the reported precision. We therefore use $r = 1$

Table 17: Calibration-state cache budgets and LLaMA-2-7B PPL. Bold marks the lowest PPL in each row, including ties at the displayed precision. Cache totals include both full-precision and quantized states.

Metric	Fusion Token	Random Token	ST $r = 0$	ST $r = 1$	ST $r = 2$	ST $r = 4$
Total cache (GiB)	4	4	2	4	6	10
W8A8 WikiText-2	5.48	5.48	**5.47**	**5.47**	**5.47**	**5.47**
W8A8 C4	**6.98**	**6.98**	**6.98**	**6.98**	**6.98**	**6.98**
W6A6 WikiText-2	5.52	5.52	5.52	**5.50**	**5.50**	**5.50**
W6A6 C4	7.02	7.02	7.02	**7.01**	**7.01**	**7.01**
W4A4 WikiText-2	5.99	5.98	6.01	**5.85**	5.86	5.87
W4A4 C4	7.66	7.65	7.67	7.55	**7.54**	7.55

to balance reconstruction quality and cache budget, and to avoid granting ST additional quantized-state storage when comparing with methods such as QuEPT and PTMQ. This comparison concerns the feature budget, not identical runtime memory across implementations.

Peak and average GPU memory and feature-fusion speed also depend on implementation, batch size, sequence length, and buffer lifetimes, so this table reports the feature-storage budget instead of cross-implementation measurements of those quantities. A smaller feature cache can potentially leave room for larger calibration batches or longer sequences and reduce the volume of data transferred between GPU and host memory. These benefits indicate the potential advantages of reducing feature-storage requirements, while the actual system-level gains depend on the specific implementation.

E.3 Rank statistics

Table 18: Allocated adapter ranks from the earlier reconstruction-capacity allocation runs over target linear layers. Ranges show minimum to maximum.

Model	W8A8	W7A7	W6A6	W5A5	W4A4
7B mean	3.37	4.79	9.92	17.73	33.74
7B range	2–17	2–20	3–30	5–53	5–58
13B mean	3.28	4.25	8.91	17.19	34.53
13B range	2–17	2–22	4–41	6–54	17–58

The pooled linear-layer means for LLaMA-2-7B are 19.49, 19.61, 15.41, 11.54, 11.51, 11.05, and 8.75 for the query, key, value, attention-output, gate, up, and down linear layers. The corresponding LLaMA-2-13B means are 18.98, 19.11, 15.43, 11.42, 11.10, 10.76, and 8.64.

F Qualitative Clipping-Error Diagnostics

The two displays below visualize clipping-induced errors at linear-layer outputs from the PrismRank rank-allocation analysis. For every candidate precision, PrismRank forms $\mathbf{E}_{l,j,b} = \overline{\mathbf{Y}}^{\mathrm{clip}}_{l,j,b} - \mathbf{Y}^{\mathrm{fp}}_{l,j}$ before computing the retained rank and singular-value mass. The saved files show the Block 0 `o_proj`, batch 0, sample 0 error from one LLaMA-3-8B rank-allocation analysis run. PrismRank aggregates the corresponding error matrices over all analysis samples and computes the same allocation score for each analyzed transformer block, linear layer, and candidate precision before the configured within-block allocation. The displays expose one source sample of the signal used for reconstruction-capacity allocation.

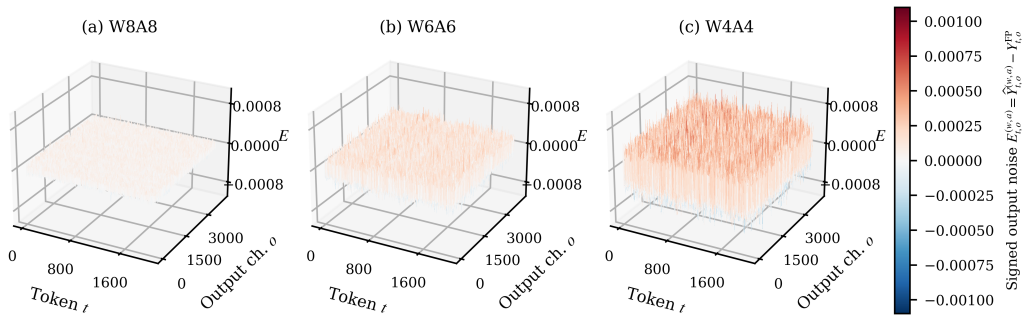


Figure 5: Signed clipping-induced error diagnostic for the archived LLaMA-3-8B Block 0 `o_proj`, batch 0, sample 0. The panels show $\mathbf{E}^{(b)}_{t,o} = \overline{\mathbf{Y}}^{\text{clip}}_{t,o,b} - \mathbf{Y}^{\text{fp}}_{t,o}$ for W8A8, W6A6, and W4A4. The renderer spatially pools the full token-by-output error map by retaining the signed entry with the largest magnitude in each bin. It provides a qualitative spatial view of the clipping-induced errors used by PrismRank for reconstruction-capacity allocation.

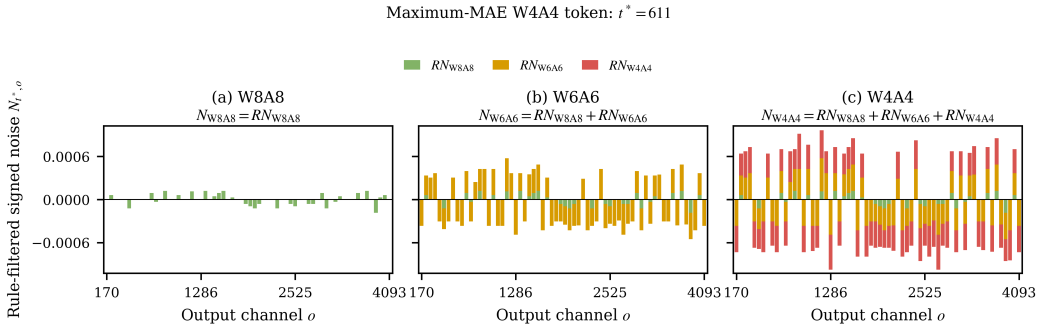


Figure 6: Rule-filtered display derived from the same clipping-induced errors. It selects $t^* = 611$, the maximum-W4A4-MAE token in the saved sample, and the 64 output channels with the largest $|\mathbf{E}^{\text{W4A4}}_{t^*,o}|$.

G Reproducibility Details

Published LLM baselines come from QuEPT Table 2, DuQuant Tables 1, 5, D8, and D10, and ABQ-LLM Tables 2, 7, 8, and 11. ABQ-LLM Avg. is the arithmetic mean of its published PIQA, ARC-E, ARC-C, HellaSwag, and WinoGrande scores. The reported DuQuant and ABQ-LLM averages exclude BoolQ. DuQuant combines rotation and scaling with channel permutation, OmniQuant combines scaling with shifts, and GPTAQ uses QuaRot preprocessing. Local PrismQ experiments use one RTX 6000 Ada GPU (SM89, 48 GiB), CUDA 12.1, and driver 550.120. Table 2 projects independent calibration over a Cartesian grid of N weight and M activation targets as tNM, where t is the single-target time. OmniQuant and QLLM coefficients divide the previously reported five-target totals by five; GPTAQ uses reported single-target means of 13, 32, and 16 minutes for LLaMA-2-7B, LLaMA-2-13B, and LLaMA-3-8B, respectively. QuEPT and PrismQ times measure joint calibration over the five evaluated diagonal formats. The current QuEPT reproduction uses WikiText-2, 128 training and 64 validation sequences of length 2048, seed 2, 15 epochs, batch size 2, MAE reconstruction, Hadamard pre-rotation, and selective feature merging. It retains its own optimization protocol; PrismQ uses seed 42 and 12 epochs. For deployment cost projection, we assume activation error dominates and an adapter calibrated at W4A4 can be reused across W*A4; off-diagonal accuracy is not evaluated here. QuEPT times use one logged training interval per model, including calibration-data preparation, block loading, reconstruction, and feature forwarding; PrismQ times include clipping pretraining and rank-allocation analysis. Evaluation and queue time are excluded. QuaRot and DuQuant require no or negligible calibration.

PrismRank uses exponent $p = 1.0$ for LLM and ViT experiments and $p = 2.0$ for VLM experiments, primarily to align singular-value aggregation with the corresponding reconstruction-loss settings. The retention threshold is fixed at $\tau = 0.9$ for all experiments. For LLM and VLM experiments, clipping pretraining uses 4 samples and PrismRank analysis uses $n_s = 2$ samples, with the sample type kept consistent with the corresponding main calibration set. For ViT, Swin, and DeiT experiments, clipping pretraining uses 64 samples and PrismRank analysis uses $n_s = 32$ samples.

The PrismQ LLaMA and Qwen2-7B PPL and accuracy runs use fake quantization for decoder linear layers and associated activations; the language-model head and final normalization remain full precision, and the key-value cache uses 16 bits. They use seed 42, batch size 2, 12 block-reconstruction epochs, the MAE objective, Hadamard pre-rotation, and an online Hadamard transform for down linear layers, with online Hadamard transforms disabled for query and key linear layers. W4 weights use group size 128, W5 through W8 use one group per output channel, activation clipping ratios are 0.8500, 0.8875, 0.9225, 0.9625, and 1.0000 for A4 through A8, and ST uses water filling with cache budget $K = n$, $\rho_{\min} = 0.05$, $\rho_{\max} = 0.5$, and uniform candidate weights. Weights use separate learned lower and upper clipping bounds; activations use per-token min–max affine quantization with the fixed clipping ratios above. AdamW uses clipping and low-rank learning rates of 0.1 and 7×10^{-5}, respectively; each block evaluates three candidates per optimizer step in a randomly offset cyclic order.

All Qwen2-7B evaluations use the Qwen2-7B-Instruct checkpoint, BF16 model storage, and the same zero-shot task metrics. The full-precision reference disables quantization and rotation. The local QuaRot run uses the same weight group sizes as PrismQ, activation clipping ratios of 1.0, and online Hadamard transforms for down, query, and key linear layers. It uses no low-rank correction or learned clipping. The VLM configuration uses LLaVA-OneVision-Qwen2-7B, abbreviated as LLaVA-OneVision-7B in QuEPT, with seed 42, rank 48, 128 COCO/ShareGPT4V image-text records (Lin et al., 2014), MSE reconstruction, ST water filling, PrismRank mixture $(0, 0.5, 0.5)$, and no pre-rotation. Clipping and low-rank learning rates are 0.2 and 10^{-4}; its vision tower and multimodal projector remain full precision.

The four PrismQ ViT runs use seed 42, 1024 ImageNet images, fake quantization, ST water filling, and PrismRank mixture $(0, 0.5, 0.5)$. Rank is 64 except for DeiT-S (128). The evaluator keeps `patch_embed.proj` at 8 bits; WbAb applies to the remaining target linear layers. Swin-S uses the optional outlier-handling skip. Table 3 uses same-device timing: ERQ is reproduced with its official TPAMI 2025 implementation and reports mean single-target calibration time; PTMQ, QuEPT, and PrismQ report joint time. PTMQ, QuEPT, and FP32 accuracy references follow QuEPT Table 1. PrismQ measures the interval from block-wise PTQ initialization through post-calibration cleanup (507, 969, 494, and 809 seconds for ViT-S, ViT-B, DeiT-S, and Swin-S), including clipping pretraining and rank-allocation analysis but excluding evaluation. FlexiQ mixes 4-bit and 8-bit channels and uses runtime bit extraction, so its 5-, 6-, and 7-bit columns indicate average weight/activation budgets rather than uniform W5A5, W6A6, and W7A7 quantization. Its first and last layers remain 8-bit.

The deployment benchmark uses the ABQ-LLM kernel framework for packed W4A4 and W8A8 GEMMs; embeddings, RMSNorm, RoPE, softmax, and the key-value cache remain FP16. Table 16 specifies sampling and warmups. NVML totals include the CUDA context of the process and runtime allocations; CUDA memory-pool counters cover only pool allocations. Brief peaks between memory samples can be missed. Each latency sample pairs separate synchronous one-token and 128-token generation calls. Their latency difference divided by 127 gives per-token decode time; the table averages the per-sample decode rates. Loading, weight conversion, first buffer allocation, validation, and tuning precede timing; dynamic activation quantization and packing remain within timed GEMM calls.

The entry point seeds Python, NumPy, PyTorch, and CUDA and requests deterministic cuDNN behavior with benchmarking disabled. PrismQ LLM runs use seed 42; local QuEPT runs use seed 2. Supplemental ledgers record result and raster provenance; checkpoint and dataset paths are machine-specific.

G.1 Allocation implementation

The reference parameter budget equals the parameter cost of assigning rank $\bar{r}$ to every target linear layer. A unit of rank at linear layer (l, j) costs

$$c_{l,j} = d_{\text{in}}^{l,j} + d_{\text{out}}^{l,j} \tag{13}$$

parameters in the two low-rank factors. PrismRank first reserves the configured minimum rank for each candidate precision, including its precision-specific portion. The remaining parameter budget is then divided between two allocation paths. One path splits its share equally across transformer blocks and distributes the budget within each block among target linear layers according to their allocation scores. The other path assigns its share uniformly across all target linear layers. Each linear layer subsequently distributes its assigned reconstruction capacity among candidate precisions according to their allocation scores.

In the main LLM configuration, the mixture weights are $(0, 0.5, 0.5)$, so the score-normalized model-level path is disabled. The two active allocation paths each receive half of the remaining parameter budget. The minimum rank per precision is 2, the precision-specific rank is 2, and the reference rank is $\bar{r} = 40$. After allocation, the assigned parameter count at each linear layer and precision is divided by $c_{l,j}$ and stochastically rounded to obtain an integer adapter rank. The resulting parameter budget is therefore approximate by construction. In the recorded reconstruction-capacity allocation study, the measured shortfall is 0.0507% for LLaMA-2-7B and 0.0550% for LLaMA-2-13B.

H Clipping Interval Nesting

H.1 Clipping contributions across precisions

Consider candidates $b_1, \ldots, b_m$ ordered from lower to higher precision, evaluated on the same weight and input tensors. For $\mathbf{T} \in \{\mathbf{W}, \mathbf{X}\}$, let $\mathbf{T}_{b_i}^c$ denote the corresponding clipped tensor. Define the highest-precision clipping error and the adjacent increments as

$$\mathbf{C}_{\text{base}}^T = \mathbf{T} - \mathbf{T}_{b_m}^c, \qquad \mathbf{H}_j^T = \mathbf{T}_{b_{j+1}}^c - \mathbf{T}_{b_j}^c. \tag{14}$$

The intermediate terms cancel, giving

$$\mathbf{T} - \mathbf{T}_{b_i}^c = \mathbf{C}_{\text{base}}^T + \sum_{j=i}^{m-1} \mathbf{H}_j^T. \tag{15}$$

Substitution into Equation 1 yields separate weight and activation contributions across precisions:

$$\begin{aligned} \mathbf{W}\mathbf{X} - \mathbf{W}_{b_i}^c \mathbf{X}_{b_i}^c = \mathbf{W} &\left(\mathbf{C}_{\text{base}}^X + \sum_{j=i}^{m-1} \mathbf{H}_j^X \right) \\ &+ \left(\mathbf{C}_{\text{base}}^W + \sum_{j=i}^{m-1} \mathbf{H}_j^W \right) \mathbf{X}_{b_i}^c. \end{aligned} \tag{16}$$

These are exact identities for clipped tensors, without rounding or low-rank correction. They do not require interval nesting. For nested intervals, the scalar clipping errors for a fixed tensor entry have consistent signs and decrease in magnitude as the interval widens. On the other hand, multiplication by weights or inputs can mix or cancel these errors, and the weight contribution is also affected by $\mathbf{X}_{b_i}^c$. Consequently, the output-space errors may contain additional error components beyond those represented by the nested clipping structure. We therefore retain precision-specific reconstruction capacity to account for these additional error components.

H.2 A Laplace model for weight clipping

Assume a fixed weight group follows $w \sim \text{Laplace}(\mu, \beta)$ with $\beta > 0$, and all precisions use the same center μ and clipping interval $[\mu - \alpha, \mu + \alpha]$. ACIQ (Banner et al., 2019) models clipping and

uniform quantization through an approximate scalar MSE. With 2^b equal bins of width $2\alpha/2^b$, its approximation is

$$D_b(\alpha) \approx \widehat{D}_b(\alpha) = 2\beta^2 e^{-\alpha/\beta} + \frac{\alpha^2}{3\,2^{2b}}. \tag{17}$$

The first term is the exact clipping MSE of the Laplace tails; the second approximates the rounding contribution. Differentiating the approximate objective gives

$$\begin{aligned} \widehat{D}_b'(\alpha) &= -2\beta e^{-\alpha/\beta} + \frac{2\alpha}{3\,2^{2b}}, \\ \widehat{D}_b''(\alpha) &= 2e^{-\alpha/\beta} + \frac{2}{3\,2^{2b}} > 0. \end{aligned} \tag{18}$$

Its unique minimizer satisfies

$$u_b e^{u_b} = 3\,2^{2b}, \qquad u_b = \frac{\alpha_b^\star}{\beta} > 0. \tag{19}$$

Since ue^u increases for $u > 0$, the optimal radius increases with b. Thus the optimal intervals in this approximate model are nested for a fixed μ, β, and grouping. This scalar model motivates the sharing structure.

H.3 Activation clipping and empirical validation

For a common activation group with minimum a and maximum c, the implemented clipping interval is $[\rho_b a, \rho_b c]$. If $a \le 0 \le c$ and $0 < \rho_b \le \rho_{b'} \le 1$, then

$$[\rho_b a, \rho_b c] \subseteq [\rho_{b'} a, \rho_{b'} c]. \tag{20}$$

The configured A4–A8 ratios increase with bit width. Interval nesting follows when the inputs are shared and the range contains zero.

H.4 Clipping pretraining statistics

A dedicated LLaMA-2-7B run measures learned weight clipping after pretraining on four WikiText-2 sequences of length 2048 with seed 42. It checks 224 target linear layers across 32 transformer blocks. Unlike the main LLM configuration, this diagnostic uses per-output-channel quantization for every weight bit width from 4 to 8, making the clipping units directly comparable. The check stops intentionally after pretraining and does not measure bounds after joint low-rank reconstruction.

Table 19 reports the raw upper- and lower-bound logits. Effective endpoints are obtained by multiplying each group's weight extrema by the sigmoid of the corresponding logit. The reported means and medians therefore describe parameters before the sigmoid, not clipping endpoints or mean clipping ratios. Counts apply separately to upper and lower logits; the pooled row covers all five precisions. The logged median uses the lower middle value when the count is even.

Table 19: Weight clipping-logit statistics after pretraining. All values retain the six decimal places provided by the log.

Bits	Count	Upper logit		Lower logit	
		Mean	Median	Mean	Median
W4	1,359,872	0.828111	0.789776	0.828111	0.789780
W5	1,359,872	1.496420	1.441197	1.496419	1.441201
W6	1,359,872	2.330540	2.256305	2.330544	2.256337
W7	1,359,872	3.599480	3.515708	3.599471	3.515690
W8	1,359,872	4.449764	4.432135	4.449776	4.432117
All	6,799,360	2.540863	2.326181	2.540864	2.326157

For a higher-precision interval $[\ell_H, u_H]$ and its adjacent lower-precision interval $[\ell_L, u_L]$, nesting requires $\ell_H \le \ell_L$ and $u_H \ge u_L$. The implementation evaluates actual endpoints, includes equality, and adds no tolerance. Table 20 aggregates matched and total counts over all linear layers rather than averaging their percentages.

Table 20: Adjacent-precision nesting of actual weight clipping intervals after pretraining. Every pair has 1,359,872 comparable intervals.

High → low	Nested	Total	Nesting (%)
W8 → W7	1,037,511	1,359,872	76.29
W7 → W6	1,210,092	1,359,872	88.99
W6 → W5	1,182,246	1,359,872	86.94
W5 → W4	1,174,858	1,359,872	86.39

I Off-Diagonal Weight–Activation Results

Table 21 evaluates all 25 combinations of W4–W8 and A4–A8 on LLaMA-2-7B. One calibration jointly fits the five diagonal formats. At evaluation, format WiAj uses the low-rank adapter trained for WjAj, while weight and activation quantization remain at i and j bits, respectively. Thus each activation precision selects one trained adapter for reuse across weight precisions; the run does not train 25 separate adapters.

The run uses WikiText-2 with 128 training and 64 validation sequences of length 2048, seed 42, 12 reconstruction epochs, asymmetric weights and activations, MAE reconstruction, Hadamard preprocessing, and ST water filling. W4 uses groups of 128 input channels; W5–W8 use per-output-channel quantization. Activation clipping ratios follow the main LLM configuration.

Table 21: LLaMA-2-7B perplexity with activation-bit adapter selection. Rows specify weight precision and columns specify activation precision. All values retain the two decimal places in the evaluation log.

	WikiText-2 PPL					C4 PPL				
Weights	**A4**	**A5**	**A6**	**A7**	**A8**	**A4**	**A5**	**A6**	**A7**	**A8**
W4	5.87	5.87	5.85	5.80	5.80	7.54	7.51	7.43	7.42	7.39
W5	5.99	5.58	5.58	5.57	5.56	7.63	7.12	7.10	7.07	7.05
W6	5.89	5.61	5.50	5.50	5.49	7.52	7.17	7.01	7.00	6.99
W7	5.86	5.60	5.52	5.48	5.48	7.49	7.15	7.02	6.98	6.99
W8	5.86	5.60	5.51	5.48	5.48	7.48	7.14	7.02	6.99	6.98