RESCOM: ACTIVATION RESIDUAL COMPENSATION FOR LARGE LANGUAGE MODEL QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-bit weight–activation quantization reduces the memory and computation needed to deploy large language models, but activation rounding can still degrade accuracy after calibration. Reconstruction from rounded inputs cannot distinguish activations within the same quantization bin. We propose ResCom, a low-rank correction that uses the activation rounding residual to compensate for the resulting layer-output error. An output-aware calibration procedure selects residual directions by their contribution to output reconstruction and computes the correction in closed form, without gradient training or updates to the calibrated weights. On Llama-3-8B with four-bit weights and activations, rank-32 attention corrections reduce WikiText-2 perplexity from 7.708 to 7.147, improve MMLU by 2.89 percentage points, and raise five-shot GSM8K accuracy from 23.96 to 36.16 percent. Evaluations on models from 1B to 70B parameters and on held-out C4 test the method beyond this setting. On B200, the evaluated FP32 implementation adds 2.25 percent to 4K prefill time; a separate compact-storage implementation retains a 27.7 percent peak-memory reduction relative to FP16.

1 INTRODUCTION

Deploying a large language model on limited hardware is constrained by the memory needed to store and move its weights and by the computation needed to generate each token. Quantization addresses these costs by representing numbers with fewer bits. When retraining is too expensive, post-training quantization (PTQ) applies this conversion to a pretrained model using a small calibration set (Xiao et al., 2023; Zhang et al., 2024). Weight-only PTQ can substantially reduce parameter storage (Frantar et al., 2023; Lin et al., 2024), but low-bit matrix multiplication also requires quantized activations. Quantizing both weights and activations, as in W4A4 inference, offers that opportunity while making accuracy harder to preserve (Ashkboos et al., 2024; Park et al., 2026). Unlike fixed weights, activations change with the input and can contain unusually large values (Xiao et al., 2023; Sun et al., 2024). With only a few bits available, covering those values increases the spacing between representable numbers. Consequently, distinct activation values can round to the same number even though the full-precision layer would map them to different outputs. The resulting loss begins at the layer input, before any quantized weight acts on it.

Existing PTQ methods approach this difficulty from two directions. The first makes activations easier to represent before rounding. SmoothQuant rescales channels to reduce activation outliers, while QuaRot and SpinQuant use rotations to spread their effect across coordinates (Xiao et al., 2023; Ashkboos et al., 2024; Liu et al., 2025). These methods reduce rounding error, but any values that still round together become indistinguishable to a subsequent computation that receives only their quantized values. The second direction compensates for errors in the quantized layer. Low-rank reconstruction methods add a compact correction to the quantized computation, and output-aware objectives help identify corrections that matter for the layer's result (Zhang et al., 2024; 2025). SERQ, for example, uses weight and activation saliency to construct an efficient reconstruction path (Park et al., 2026). Its inference-time correction nevertheless acts on rounded activations. If two inputs have become the same quantized vector, changing the weights applied to that vector cannot restore the different outputs the original layer would have produced.

The information needed to distinguish those inputs is their *activation rounding residual*: the difference between an activation and its quantized value. This residual is available when quantization

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.

takes place, but using it in a full-size, high-precision matrix multiplication would undermine the savings of the low-bit path. A compact correction must therefore retain only the residual directions that matter most. Their importance cannot be judged from residual magnitude alone: a small discrepancy in one direction may change the layer output more than a large discrepancy in another. This leads to our central question: can a limited-rank branch use the information discarded at activation quantization to recover the calibrated layer's output, while leaving its low-bit main computation intact?

We propose RESCOM, a post-calibration correction that answers this question by feeding activation rounding residuals to a low-rank branch. Calibration examples provide the output errors of the frozen quantized layer; a closed-form fit jointly determines which residual combinations to retain and how they should correct those errors. The main quantized path and its existing scaling, rotations, or weight reconstruction remain in place. In this way, RESCOM targets input-specific information loss at the activation quantizer without gradient-based retraining.

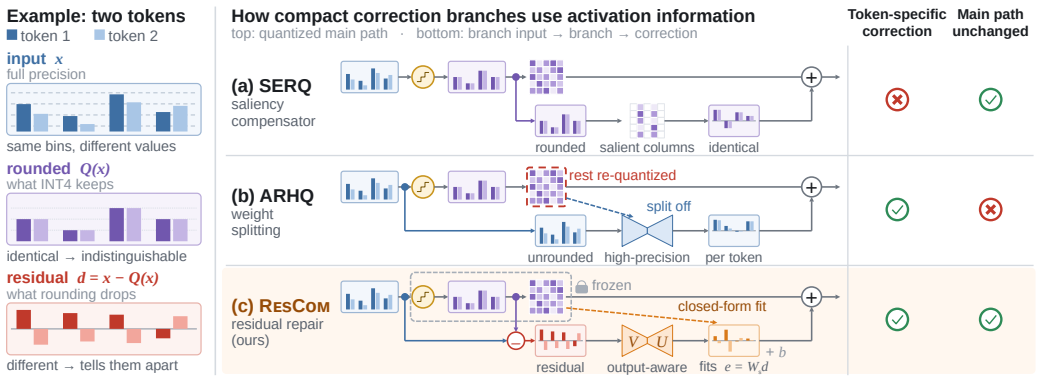


Figure 1: Three reconstruction paths. **(a)** SERQ applies a selected-channel compensator to rounded activations. **(b)** ARHQ splits weights before quantization and uses unrounded activations. **(c)** ResCom instead feeds the activation rounding residual to an output-aware low-rank branch fitted after calibration. The residual preserves token-specific distinctions within each quantization bin, while post-calibration fitting leaves the host weights and quantizer unchanged. Section 3 and Appendices B–C formalize these differences.

Our contributions are threefold:

- We propose **residual-conditioned activation repair**, which supplies an auxiliary branch with the input information lost during rounding while preserving the calibrated quantized path.
- We develop **output-aware residual selection**, which identifies residual directions that explain layer-output error and fits a low-rank correction analytically, without backpropagation or weight updates.
- Experiments show **accuracy recovery across model scales**: Llama-3-8B MMLU improves by 2.89 points and GSM8K by 12.2 points; LLaMA-2-70B zero-shot accuracy rises from 69.59% to 70.00%. Matched-capacity controls examine the role of the residual input.

2 RELATED WORK

Low-bit representations and precision allocation. Methods for low-bit LLM inference differ in where they address quantization error: some allocate precision to sensitive values, others reshape activations before rounding, and others reconstruct the resulting output. Precision allocation began with weight-only PTQ. GPTQ uses calibration data to compensate for sequential weight-rounding errors (Frantar et al., 2023). AWQ further observes that a small set of channels identified from activations deserves particular protection and rescales their weights accordingly (Lin et al., 2024). Both reduce weight storage, but their matrix products still use high-precision activations. Quantizing activations too makes low-bit arithmetic possible, while exposing it to activation outliers. To preserve these values, Dettmers et al. separate outlier dimensions for high-precision multiplication in LLM.int8() (Dettmers et al., 2022). Atom moves more of the computation to low-bit operators by grouping outlier channels into a regular mixed-precision serving path (Zhao et al., 2024). Such designs preserve exceptional values at the cost of higher-precision work; values assigned to the low-bit

path still incur rounding error. This motivates approaches that make activations easier to quantize in the first place.

Activation distribution shaping. Rather than assigning outliers a separate precision, activation shaping reduces their effect on the quantization range. SmoothQuant rescales channels so that activation outliers are partly transferred into the weights (Xiao et al., 2023). Because this transfer can make weight quantization harder, OmniQuant learns an equivalent transformation together with weight clipping from calibration data (Shao et al., 2024). Channel-wise scaling has limited freedom to redistribute values across coordinates, which motivates a change of basis. QuaRot uses output-preserving rotations to spread activation outliers (Ashkboos et al., 2024). As the effectiveness of fixed rotations can vary across layers, SpinQuant learns rotations against quantization error (Liu et al., 2025). FlatQuant extends this approach with learned affine transformations to flatten distributions that remain uneven after scaling or rotation (Sun et al., 2025). Outliers can also depend on token position rather than channel alone; PrefixQuant addresses these by adding prefixed tokens that isolate token-wise outliers (Chen et al., 2025). Each method changes what reaches the quantizer, but distinct values can still round to the same code afterward. Recovering their different effects on the layer output calls for a correction after rounding.

Low-rank reconstruction and calibration. Low-rank reconstruction adds a compact correction for error that quantization has already introduced. LQER uses activation-derived scaling to make weight quantization error more amenable to low-rank approximation (Zhang et al., 2024). Approximating that weight error, however, does not necessarily minimize the error at the layer output; QERA addresses this mismatch by deriving a correction from an output-error objective (Zhang et al., 2025). Conventional two-factor corrections require sequential products and, on a fully low-bit path, an intermediate that must also be represented or quantized. SERQ addresses this execution problem directly: it selects salient weight rows for reconstruction and combines static activation flattening with offline permutation, enabling a single auxiliary low-bit product in W4A4 inference (Park et al., 2026). Its auxiliary product still receives rounded activations, so inputs mapped to the same quantized vector receive the same correction. ARHQ recognizes the importance of activation residuals, using their covariance to choose a high-precision low-rank weight branch before quantizing the remaining weights (Wang et al., 2026). RESCOM instead starts from an already calibrated quantized layer and uses each input's rounding residual to fit a low-rank correction to its output error.

3 METHOD

The preceding section identifies a limit of corrections that receive only rounded activations: they cannot distinguish inputs mapped to the same quantized value. ResCom addresses this limit after the original quantization pipeline has been calibrated. It passes the information removed by activation rounding to a small correction branch, then fits that branch to the missing layer output. The quantized path and any existing weight reconstruction remain frozen throughout calibration. Figure 2 shows how the branch is fitted and used.

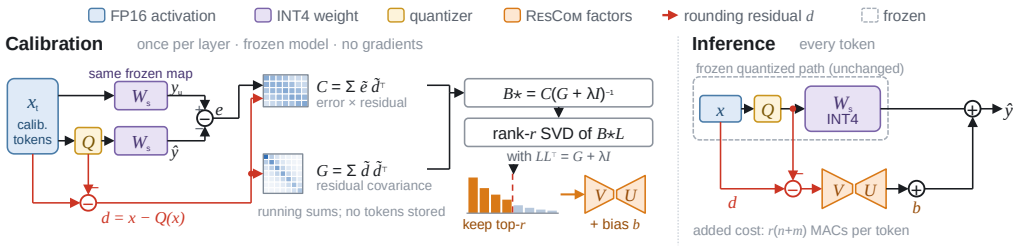


Figure 2: ResCom calibration and inference. Calibration pairs each activation rounding residual with the output contribution removed by rounding and fits a low-rank correction. At inference, the branch evaluates that correction for the current token and adds it to the frozen quantized output. The shared layer bias is omitted.

3.1 RESIDUAL-CONDITIONED CORRECTION

Recovering the missing input distinction. Two activations can have the same quantized value while occupying different positions within its quantization bin. A correction computed only from that value must be identical for both. To distinguish them, the branch needs a signal that changes with the position lost during rounding. It must also obtain that signal when each token is processed, since the position changes with the input. Both conditions are met by the *rounding residual*: the difference between the activation entering the quantizer and its dequantized output. This difference is available at the quantizer and requires neither task labels nor another model pass.

Defining the correction target. Let $x \in \mathbb{R}^n$ be the activation in the coordinate system of a calibrated input quantizer, and let $Q(x)$ be its quantize–dequantize output. We define the residual as $d = x - Q(x)$. To measure only the output lost through activation rounding, we apply the same frozen linear map W_s to both inputs. This map includes the quantized weights and any weight-reconstruction term already supplied by the host method; it need not be stored as a new dense matrix. With the existing bias c,

$$y_u = W_s x + c, \qquad \hat{y} = W_s Q(x) + c, \qquad e = y_u - \hat{y} = W_s d. \tag{1}$$

Thus, e is the output contribution removed by rounding in the *already calibrated* layer. Using the same W_s on both sides prevents remaining weight-approximation error from entering this target.

Adding a limited-rank branch. Computing $W_s d$ directly at inference would add a dense multiplication. ResCom instead learns a rank-r approximation with factors $V \in \mathbb{R}^{r\times n}$ and $U \in \mathbb{R}^{m\times r}$, where m is the output width. An intercept $b \in \mathbb{R}^m$ captures an output offset without using the rank budget. The repaired layer computes

$$\hat{y}_{\mathrm{RC}} = \hat{y} + U(Vd) + b. \tag{2}$$

The factors are fixed after calibration, while d changes with each token. Choosing those factors is therefore a question of which residual variations matter most to the output.

3.2 OUTPUT-AWARE CLOSED-FORM CALIBRATION

The residual identifies what rounding removed, but its largest variations need not produce the largest output errors: the frozen layer may suppress some directions and amplify others. Given T calibration pairs (d_t, e_t) from Eq. (1), ResCom selects directions according to the output they recover. Writing the branch matrix as $B = UV \in \mathbb{R}^{m\times n}$, it solves

$$\min_{\mathrm{rank}(B)\le r,\, b} \sum_{t=1}^{T} \|e_t - Bd_t - b\|_2^2 + \lambda \|B\|_F^2, \qquad \lambda > 0. \tag{3}$$

The first term assigns rank to reductions in layer-output error. The ridge term stabilizes the fit when calibration data provide little evidence about a direction. The unpenalized intercept absorbs the mean error, leaving the low-rank factors to model differences between tokens.

Fitting from calibration statistics. ResCom collects residuals and their target output errors while running the *frozen* model on calibration data. Removing their means separates the constant output offset from the token-dependent correction. Define

$$\begin{aligned} &\bar{d} = T^{-1}\textstyle\sum_t d_t, \quad \bar{e} = T^{-1}\sum_t e_t, \quad \widetilde{d}_t = d_t - \bar{d}, \quad \widetilde{e}_t = e_t - \bar{e}, \\ &G = \textstyle\sum_t \widetilde{d}_t \widetilde{d}_t^{\top}, \qquad C = \sum_t \widetilde{e}_t \widetilde{d}_t^{\top}. \end{aligned} \tag{4}$$

Here, $G \in \mathbb{R}^{n\times n}$ records how residual directions occur together, while $C \in \mathbb{R}^{m\times n}$ records how they relate to output error. These statistics can be accumulated without retaining every token.

To rank directions by their contribution to Eq. (3), factor $G + \lambda I_n = LL^{\top}$ and compute $CL^{-\top} = \Phi\Sigma\Psi^{\top}$. If the subscript r denotes the leading r singular components, the fitted branch is

$$U = \Phi_r \Sigma_r, \qquad V = \Psi_r^{\top} L^{-1}, \qquad b = \bar{e} - UV\bar{d}. \tag{5}$$

This keeps the directions that explain the greatest reduction in the regularized output-error objective (Izenman, 1975). Appendix A gives the derivation and numerical details.

All layer statistics are collected before any fitted branch is attached. Otherwise, an early correction would change the activations observed by later layers, so their calibration targets would no longer describe the same frozen model. Algorithm 1 summarizes this order.

Algorithm 1 Closed-form calibration of ResCom

Input: Frozen calibrated model f_θ with fixed parameters θ; calibration set $\mathcal{D}$; target layers $\mathcal{L}$; rank r; ridge coefficient λ

Output: Model f_θ augmented with one ResCom branch per target layer

1: Run $\mathcal{D}$ through frozen f_θ; stream the moments of (d_t, e_t) for every $\ell \in \mathcal{L}$
2: **for** each target layer $\ell \in \mathcal{L}$ **do**
3: Compute $\bar{d}_\ell$, $\bar{e}_\ell$, G_ℓ, and C_ℓ using Eq. (4)
4: Compute $L_\ell L_\ell^\top = G_\ell + \lambda I_n$, where I_n is the $n \times n$ identity
5: Compute the rank-r SVD $C_\ell L_\ell^{-\top} \approx \Phi_{\ell,r} \Sigma_{\ell,r} \Psi_{\ell,r}^\top$
6: Recover U_ℓ, V_ℓ, and b_ℓ using Eq. (5)
7: **end for**
8: Attach all fitted branches to f_θ using Eq. (2)
9: **return** f_θ

3.3 Distinction from SERQ and ARHQ

SERQ, ARHQ, and ResCom all use compact branches to limit quantization error, but they assign those branches different inputs and change the main path at different stages. Table 1 makes the distinction explicit.

Table 1: How the three methods use activation information in their correction branches.

Method	Branch input	Main-path treatment	Use of rounding residual
SERQ	Rounded activation	Constructs a saliency-aware weight correction	No token-specific residual input
ARHQ	Unrounded activation	Splits weights and quantizes the remaining main-path weights	Residual statistics select the weight subspace
ResCom	Rounding residual	Keeps the calibrated main path frozen	Residual directly predicts each token's missing output

SERQ uses activation and weight saliency to construct a single auxiliary low-bit product, avoiding the sequential products of conventional low-rank reconstruction (Park et al., 2026). Because that product receives the rounded activation, it cannot give different corrections to two inputs with the same quantized vector. ARHQ recognizes that activation residuals reveal which weight directions amplify rounding error. It uses residual statistics to select a high-precision weight branch, then quantizes the weights remaining on the main path; its branch is evaluated on the unrounded activation (Wang et al., 2026). ResCom takes a different attachment point. It leaves the calibrated weights and quantizer intact and fits a branch on each token's residual, using the frozen layer's missing output as supervision. The distinction is thus both *when* the correction is fitted and *which input information* reaches it.

3.4 Integration and cost

ResCom attaches at the input quantizer of a chosen linear projection. It forms the residual before the unrounded activation is discarded and adds the branch output after the existing linear computation. Scaling, rotation, or permutation introduced by the host pipeline is reflected in the coordinate system of x; ResCom does not need to undo those transformations. The branches can be attached to selected attention or feed-forward projections after calibration. The default configuration repairs attention projections, and Section 4.3 evaluates other placements.

For an n-input, m-output layer, calibration stores $O(n^2 + mn)$ statistics, independent of the number of calibration tokens; layers can be processed in groups to limit memory. This work is performed once, without gradients or updates to the quantized weights. At inference, forming the residual

requires an n-element subtraction, and the branch uses $r(n+m)$ multiply-accumulates per token. Its factors and intercept store $r(n+m)+m$ scalars. The chosen rank therefore sets both the capacity of the output correction and its added inference cost.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models and comparisons. We evaluate Llama-3/3.2 1B–8B, LLaMA-2 7B/13B/70B, Mistral-7B-v0.3, and Gemma-3-1B/4B (Touvron et al., 2023; Grattafiori et al., 2024; Jiang et al., 2023; Gemma Team, Google DeepMind, 2025). SERQ provides the main quantized checkpoints; every ResCom result is paired with our evaluation before repair. PrefixQuant and FlatQuant test transfer to other pipelines. W4A4 is the main setting; W4A8 appears in Appendix D.2. Within each pair, the checkpoint, tokenizer, evaluation harness, and test examples are fixed, so the comparison isolates the fitted branch rather than a change in quantization or evaluation protocol.

Table 2: W4A4 WikiText-2 PPL and task accuracy. †: published by SERQ (Park et al., 2026); other rows are our evaluations. ‡: nominal factor cost. Zero-shot averages eight tasks; bold compares paired rows.

		#Eff.	Train.	MAC	LLaMA-2 7B			LLaMA-2 70B			LLaMA-3 8B		
#Bits	Method	bits/wt.	free	overhead	PPL↓	0-shot↑	MMLU↑	PPL↓	0-shot↑	MMLU↑	PPL↓	0-shot↑	MMLU↑
FP16	Full precision†	16	✓	–	5.47	64.09	41.83	3.53	70.54	65.44	6.13	67.16	62.13
W4A4	SmoothQuant†	∼4	✓	–	1.51e4	35.44	26.04	–	–	–	4.71e5	36.34	25.17
	SmoothQuant, g128†	4.13	✓	–	7.49	57.15	30.40	–	–	–	17.26	48.97	29.30
	QuaRot†	∼4	✓	19.8%	6.15	59.53	33.58	–	–	–	8.41	59.12	47.29
	SpinQuant†	∼4	×	19.8%	6.00	61.00	34.80	–	–	–	8.26	61.75	49.93
	SERQ	4.24	✓	18.7%	5.939	**62.52**	37.49	3.638	69.59	63.72	7.708	62.98	54.69
	RESCOM	4.30‡	✓	0.39%‡	**5.847**	62.30	**38.25**	**3.597**	**70.00**	**64.16**	**7.147**	**64.89**	**57.58**

		#Eff.	Train.	MAC	LLaMA-2 13B			LLaMA-3.2 1B			LLaMA-3.2 3B		
#Bits	Method	bits/wt.	free	overhead	PPL↓	0-shot↑	MMLU↑	PPL↓	0-shot↑	MMLU↑	PPL↓	0-shot↑	MMLU↑
FP16	Full precision†	16	✓	–	4.88	66.53	52.04	9.75	54.82	36.76	7.81	62.66	54.06
W4A4	SmoothQuant†	∼4	✓	–	1.32e4	34.53	23.85	1.53e5	35.59	24.37	3.73e4	35.72	23.41
	SmoothQuant, g128†	4.13	✓	–	6.31	61.28	39.83	69.22	40.04	24.43	53.33	44.17	27.31
	QuaRot†	∼4	✓	19.8%	5.41	62.55	47.25	13.17	50.03	26.64	9.73	55.76	44.75
	SpinQuant†	∼4	×	19.8%	5.20	64.80	47.80	13.47	48.95	26.38	10.15	56.88	42.42
	SERQ	4.24	✓	18.7%	5.227	66.08	49.20	12.461	50.93	31.05	9.177	58.98	47.96
	RESCOM	4.30‡	✓	0.39%‡	**5.148**	**66.22**	**49.35**	**12.071**	**51.02**	**31.21**	**8.964**	**60.14**	**49.29**

Calibration and metrics. The rank-32 attention branch uses 128 WikiText-2 windows of 2048 tokens and $\lambda = 0.01\,\mathrm{tr}(G)/n$. We report WikiText-2 and held-out C4 PPL, an eight-task zero-shot mean, 57-subject MMLU, five-shot GSM8K, and three LongBench tasks (Merity et al., 2017; Raffel et al., 2020; Hendrycks et al., 2021; Cobbe et al., 2021; Bai et al., 2024). Protocols are in Appendix F (Gao et al., 2024).

4.2 MAIN COMPARISON

Accuracy after residual repair. Tables 2–3 show lower PPL across all evaluated backbones. On Llama-3-8B, PPL falls from 7.708 to 7.147, MMLU gains 2.89 points, and zero-shot gains 1.91 points; Gemma-3-4B MMLU gains 1.62 points. LLaMA-2-70B also improves in PPL and both accuracy aggregates. Across these paired results, PPL improves consistently from 1B to 70B, while Llama-3-8B and Gemma-3-4B show the largest downstream gains.

Table 3: Cross-family controls and transfer. **(a)** Unquantized precision is FP16 for Mistral and BF16 for Gemma; W4A4 is direct rounding without an added algorithm. Accuracy is in percent; zero-shot averages eight tasks and MMLU covers 57 subjects. **(b)** C4 is unseen during WikiText-2 fitting; ρ_{32} is the mean layerwise fit reduction from Eq. (14). Bold compares each SERQ/ResCom pair.

(a) Cross-family accuracy

Model	Method	WT2	C4	0-shot	MMLU
Mistral-7B v0.3	FP16	5.522	8.480	68.77	59.66
	W4A4	5.77e3	3.69e3	35.32	24.43
	SERQ	5.663	8.244	67.10	56.12
	ResCom	**5.617**	**8.188**	**67.13**	**56.73**
Gemma-3-1B PT	BF16	14.088	19.980	56.69	24.81
	W4A4	1.68e4	1.36e4	35.50	23.99
	SERQ	17.332	23.895	53.02	24.60
	ResCom	**17.268**	**21.850**	**53.17**	**24.80**
Gemma-3-4B PT	BF16	10.695	16.755	66.63	57.83
	W4A4	8.47e4	6.40e4	35.22	24.75
	SERQ	12.522	17.347	62.12	50.55
	ResCom	**12.191**	**16.557**	**62.96**	**52.17**

(b) Paired corpus transfer

	WikiText-2		C4		
Backbone	SERQ	ResCom	SERQ	ResCom	ρ_{32}
Llama-3-8B	7.708	**7.147**	11.192	**10.349**	.294
LLaMA-2 7B	5.939	**5.847**	7.432	**7.315**	.271
LLaMA-2 13B	5.227	**5.148**	6.693	**6.634**	.270
Llama-3.2-3B	9.177	**8.964**	12.688	**12.317**	.282
Llama-3.2-1B	12.461	**12.071**	17.227	**16.712**	.355
Mistral-7B-v0.3	5.663	**5.617**	8.244	**8.188**	.343

Direct W4A4 rounding substantially degrades all three additional backbones, whereas SERQ restores usable performance. ResCom then lowers WikiText-2 and C4 PPL on all three; Mistral improves all four aggregates, Gemma-3-4B gains 0.84 zero-shot and 1.62 MMLU points, and Gemma-3-1B lowers C4 from 23.895 to 21.850.

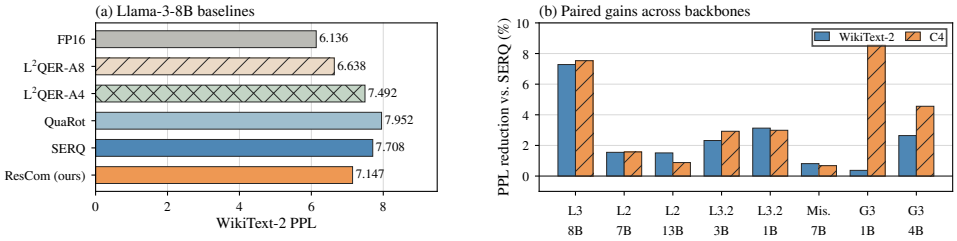


Figure 3: Perplexity comparisons. **(a)** Reproduced Llama-3-8B baselines; L^{2}QER uses MXINT and QuaRot/SERQ use INT4 simulation. **(b)** Paired W4A4 reductions across backbones. Table 2 identifies published values.

Transfer beyond the calibration setting. Table 4 shows transfer in two directions. A WikiText-2 fit lowers held-out C4 PPL on all five Llama backbones without refitting, including 11.192 to 10.349 on Llama-3-8B. The same construction improves SERQ, PrefixQuant, and FlatQuant from 7.708/7.860/6.972 to 7.147/7.605/6.960. Thus the branch transfers across corpora and attaches after distinct host pipelines; targets are defined in Appendix C.

Table 4: Transfer tests. **(a)** C4 PPL after WikiText-2 calibration, without refitting. **(b)** WikiText-2 PPL on Llama-3-8B under three host pipelines; only the correction changes within each pair.

(a) Cross-corpus transfer

Backbone	Host	ResCom	Backbone	Host	ResCom
Llama-3-8B	11.192	**10.349**	Llama-3.2-3B	12.688	**12.317**
LLaMA-2 7B	7.432	**7.315**	Llama-3.2-1B	17.227	**16.712**
LLaMA-2 13B	6.693	**6.634**			

(b) Cross-pipeline transfer

Pipeline / sites	Host	ResCom
SERQ / attention	7.708	**7.147**
PrefixQuant / output	7.860	**7.605**
FlatQuant / output+down	6.972	**6.960**

Reasoning and long-context generation. Strict GSM8K rises from 23.96% to 36.16% on Llama-3-8B and from 13.42% to 16.91% on Llama-3.2-3B; both improve on all three LongBench tasks (Table 5). Across the two Gemma models, seven of the ten reported metrics improve: Gemma-3-1B improves all three LongBench tasks, while Gemma-3-4B improves GSM8K, QMSum, and RepoBench-P. The gains span reasoning, summarization, dialogue, and code completion.

Table 5: W4A4 reasoning and long-context generation. †: published SERQ values (Park et al., 2026); other SERQ rows are our paired evaluations. GSM8K uses five shots; LongBench uses full sets with a 7500-token budget.

			GSM8K 5-shot ↑		LongBench ↑		
Model	Bits	Method	Flexible	Strict	QMSum	SAMSum	RepoBench-P
Llama-3-8B	FP16	Full precision†	48.07	47.69	23.43	46.26	66.56
	W4A4	L^2QER†	7.96	7.66	0.09	1.98	5.65
	W4A4	SERQ	24.56	23.96	21.12	42.31	61.03
		ResCom	**36.54**	**36.16**	**22.96**	**44.40**	**65.04**
Llama-3.2-3B	FP16	Full precision†	26.08	25.93	23.94	42.98	64.42
	W4A4	L^2QER†	3.56	2.73	14.92	33.87	41.99
	W4A4	SERQ	13.80	13.42	20.67	41.03	62.39
		ResCom	**17.21**	**16.91**	**21.72**	**41.17**	**62.64**
Gemma-3-1B-PT	W4A4	SERQ	**2.12**	**1.36**	14.15	6.30	42.78
		ResCom	1.82	0.99	**15.10**	**7.33**	**43.92**
Gemma-3-4B-PT	W4A4	SERQ	16.22	16.00	16.25	**16.73**	53.36
		ResCom	**23.05**	**22.82**	**20.75**	8.17	**59.36**

4.3 Ablation and Sensitivity Analysis

Table 6: Matched-rank input ablation. Each feature uses the same target and relative ridge rule. Columns give PPL and mean normalized regularized fit reduction ρ_r (Appendix A).

	rounded act.	unrounded activation		rounding residual	
rank	PPL	PPL	ρ_r	PPL	ρ_r
8	8.210	7.241	0.015	**7.203**	0.147
16	8.299	7.237	0.018	**7.167**	0.211
32	8.327	7.242	0.021	**7.147**	0.294
64	8.362	7.235	0.025	**7.128**	0.387
128	8.384	7.227	0.028	**7.110**	0.493
256	8.417	7.229	0.032	**7.085**	0.623
Mistral-7B-v0.3, one rank; unrepaired 5.663					
32	5.787	5.652	0.028	**5.617**	0.343

Table 7: Attention rank sensitivity on Llama-3.2 at W4A4. Rank zero is unmodified.

Rank	1B	3B
0	12.46	9.18
8	12.22	9.03
16	12.16	9.02
32	12.07	8.96
64	11.99	8.92
128	11.91	8.90
256	11.81	8.87

Which input should the correction use? At fixed rank, sites, target, and fit, the residual input beats the unrounded input at every rank and the rounded input by a wide margin (Table 6); Mistral follows the same order. The rounded input loses within-bin distinctions, whereas the unrounded activation mixes the retained $Q(x)$ with the missing d. Feeding d isolates information absent from the main path, explaining the matched-capacity advantage.

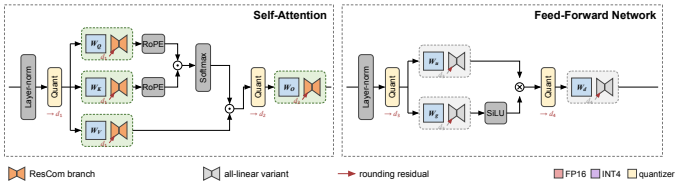


Figure 4: Repair sites. Orange marks default attention branches; gray marks added feed-forward branches. Residuals d_1–d_4 follow the quantizers; Q, K, V, O and u, g, d label the projections.

Table 8: Calibration stability and repair scope. **(a)** Mean ΔPPL with sample s.d. across fixed-evaluation calibration draws. **(b)** Llama-3-8B choices; the channel control matches the default branch's parameter and factor-MAC budgets.

(a) Calibration stability

Backbone	Draws	Δ PPL
Llama-3-8B	4	0.5604 ± 0.0059
LLaMA-2 7B	3	0.0916 ± 0.0030
Mistral-7B-v0.3	4	0.0431 ± 0.0024

(b) Repair configuration

Repaired sites	PPL ↓	Repaired sites / basis	PPL ↓
Bias only	7.715	Attention (default)	7.147
Output	7.672	All linear	**7.070**
Down	7.624	Budget-matched channels	**7.135**
Gate + up	7.688		

Where and how much to repair. Attention-only repair provides most of the gain; all-linear repair reaches 7.070 PPL with a larger budget (Figure 4, Table 8(b), and Appendix Table 11). Rank gives diminishing gains on Llama-3.2 (Table 7). The budget-matched selected-channel branch reaches 7.135 versus 7.147 for dense rank 32; comparable performance across bases identifies residual conditioning as their shared source of improvement (Appendix A.2).

Sensitivity to calibration data. Across repeated calibration draws, PPL reductions keep the same sign and sample standard deviations stay below 0.006 (Table 8(a); Appendix Table 12). On Llama-3-8B, 32/128/512 calibration windows yield 7.139/7.147/7.137 PPL (Appendix Table 13). Appendices A and D.2 relate ρ_r to absolute error scale and downstream accuracy.

4.4 Efficiency and Implementation Analysis

We separately measure FP32 simulation, compact storage, and native MXFP4; Appendix G gives kernel details.

Prefill and decoding latency on B200. In FP32 simulation (Figure 5, left), ResCom adds 2.25% to 4K prefill (1342.32 to 1372.46 ms) while PPL falls from 7.708 to 7.147; cached decoding rises from 76.79 to 86.11 ms/token.

Peak memory and accuracy with compact weight storage. A separate compact implementation (Figure 5, right) uses 14.233 GiB versus 19.686 GiB for FP16 at 4096+32 tokens. The branch adds 0.103 GiB (0.73%) over SERQ while improving MMLU and zero-shot accuracy.

Native low-bit execution. Native MXFP4 repair lowers full WikiText-2 PPL by 0.132 across 224 projections, adding 0.103 GiB and 3.7% latency over plain MXFP4 (Appendix Table 20). Both variants use less memory and take longer than BF16.

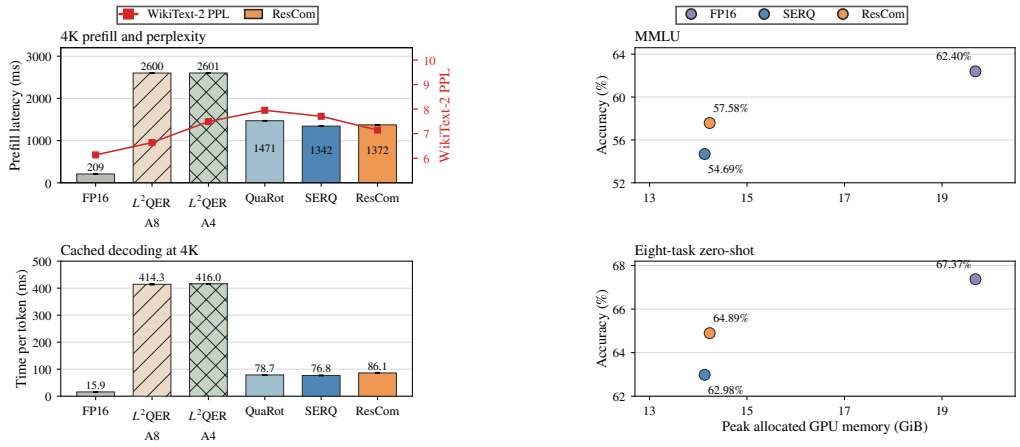


Figure 5: B200 efficiency for Llama-3-8B. **Left:** FP32-simulated latency. **Right:** compact-storage memory and accuracy; panels use separate implementations.

5 Conclusion

Low-bit activation quantization creates token-specific output errors that are no longer identifiable from rounded activations alone. We introduced ResCom, a low-rank branch conditioned on the rounding residual and fitted to layer-output error by closed-form reduced-rank regression, while leaving the calibrated weights and main quantized path unchanged. Across the evaluated 1B to 70B backbones, the correction consistently lowers paired perplexity; on Llama-3-8B it reduces WikiText-2 PPL from 7.708 to 7.147, raises MMLU by 2.89 points, and improves five-shot GSM8K from 23.96% to 36.16%. Matched-input, cross-corpus, and cross-pipeline studies connect these gains to the residual conditioning signal, while deployment measurements quantify the additional latency and memory. The evidence supports activation residuals as a useful complementary signal for repairing an already calibrated low-bit model, with the attainable gain depending on the backbone, task, and quantization path.

AI USE STATEMENT

Generative AI tools assisted with research discussion, experiment planning, code implementation, figure preparation, prose editing, and checks of reported results. The authors reviewed the assisted work against source code, raw outputs, and cited literature and take responsibility for the paper's content.

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, 2024.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, 2024. doi: 10.18653/v1/2024.acl-long.172.

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

Mengzhao Chen, Yi Liu, Jiahao Wang, Yi Bin, Wenqi Shao, and Ping Luo. PrefixQuant: Eliminating outliers by prefixed tokens for large language models quantization. In *International Conference on Learning Representations*, 2025.

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*, 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, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

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, Stella Biderman, Sid Black, et al. A framework for few-shot language model evaluation, 2024. Zenodo, version 0.4.3, `https://doi.org/10.5281/zenodo.12608602`.

Gemma Team, Google DeepMind. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Alan Julian Izenman. Reduced-rank regression for the multivariate linear model. *Journal of Multivariate Analysis*, 5(2):248–264, 1975. doi: 10.1016/0047-259X(75)90042-1.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

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, 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: LLM quantization with learned rotations. In *International Conference on Learning Representations*, 2025.

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

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

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

Yeonsik Park, Hyeonseong Kim, and Seungkyu Choi. SERQ: Saliency-aware low-rank error reconstruction for LLM quantization. In *International Conference on Learning Representations*, 2026.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 4463–4473, 2019.

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

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *First Conference on Language Modeling*, 2024.

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

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

YiFeng Wang, Zhun Sun, and Keisuke Sakaguchi. Technical report: Activation residual hessian quantization (ARHQ) for low-bit LLM quantization. *arXiv preprint arXiv:2605.00140*, 2026.

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 *PMLR*, 2023.

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

Cheng Zhang, Jianyi Cheng, George A. Constantinides, and Yiren Zhao. LQER: Low-rank quantization error reconstruction for LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *PMLR*, pp. 58763–58779, 2024.

Cheng Zhang, Jeffrey T. H. Wong, Can Xiao, George A. Constantinides, and Yiren Zhao. QERA: An analytical framework for quantization error reconstruction. In *International Conference on Learning Representations*, 2025.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate LLM serving. In *Proceedings of Machine Learning and Systems*, volume 6, 2024.

APPENDIX

A CLOSED-FORM SOLUTION AND FIT DIAGNOSTICS

Let $J(B, b)$ denote the objective in Eq. (3), with $B = UV$. Define

$$\bar{d} = \frac{1}{T}\sum_{t=1}^{T} d_t, \qquad \bar{e} = \frac{1}{T}\sum_{t=1}^{T} e_t,$$

and let $\widetilde{d}_t = d_t - \bar{d}$ and $\widetilde{e}_t = e_t - \bar{e}$. Minimizing over the unpenalized bias first gives

$$b = \bar{e} - B\bar{d}. \tag{6}$$

The input-dependent fit is described by the centered residual covariance and residual–error cross-covariance

$$G = \sum_{t=1}^{T} \widetilde{d}_t \widetilde{d}_t^{\top}, \qquad C = \sum_{t=1}^{T} \widetilde{e}_t \widetilde{d}_t^{\top}. \tag{7}$$

These statistics can be accumulated in a streaming pass, so individual calibration vectors need not be retained.

Define the centered target energy $E = \sum_{t=1}^{T} \|\widetilde{e}_t\|_2^2$ and $G_\lambda = G + \lambda I_n$. After eliminating the intercept, the remaining objective is

$$J(B, \bar{e} - B\bar{d}) = E - 2\,\mathrm{tr}(BC^{\top}) + \mathrm{tr}(BG_\lambda B^{\top}). \tag{8}$$

Here tr is the matrix trace. Since $\lambda > 0$, G_λ is positive definite, and the unrestricted ridge solution is

$$B_\star = CG_\lambda^{-1}. \tag{9}$$

Factor $G_\lambda = LL^{\top}$ with a lower-triangular Cholesky factor L. Completing the square gives

$$J(B, \bar{e} - B\bar{d}) = E - \|B_\star L\|_F^2 + \|(B - B_\star)L\|_F^2. \tag{10}$$

The invertible change of variables $Z = BL$ preserves rank. Moreover,

$$B_\star L = CL^{-\top}. \tag{11}$$

Let the compact singular value decomposition be

$$CL^{-\top} = \Phi\Sigma\Psi^{\top}, \tag{12}$$

with singular values in decreasing order. Retaining the leading r components gives the reduced-rank solution

$$U = \Phi_{:,1:r}\Sigma_{1:r,1:r}, \qquad V = \Psi_{:,1:r}^{\top}L^{-1}, \qquad b = \bar{e} - UV\bar{d}. \tag{13}$$

Thus the rank constraint keeps the directions that explain the greatest reduction in the regularized output-error objective, rather than those with the largest residual magnitude. This is the reduced-rank regression construction (Izenman, 1975) under a ridge-adjusted metric.

In implementation, neither G_{λ}^{-1} nor L^{-1} is formed explicitly. Cholesky and triangular solves are used to obtain $CL^{-\top}$ and V, improving numerical stability.

Let σ_i, $i = 1, \ldots, s$, be the ordered singular values on the diagonal of Σ, and let J_r be the minimum objective at rank r. For $E > 0$, define the normalized regularized fit reduction ρ_r by

$$\rho_r = \frac{\sum_{i=1}^{r}\sigma_i^2}{E}, \qquad J_r = E(1-\rho_r), \qquad 0 \le \rho_r \le 1. \tag{14}$$

The numerator is the regularized objective reduction from the retained singular components. The bounds follow from nonnegativity of the objective and the squared singular values. The penalty acts on B, not independently on U, V. The unpenalized calibration error reduction of this fitted map is $E\rho_r + \lambda\|B\|_F^2$; ρ_r therefore describes the regularized objective. Reported model-level values are arithmetic means over repaired layers, not energy-weighted global bounds. If $E = 0$, the ratio is undefined; the constant bias $b = \bar{e}$ suffices on the calibration samples. Tied singular values may make the rank-constrained minimizer nonunique.

A.1 Error scale and held-out correction

Let $\bar{y}_{\mathrm{ref}} = T^{-1}\sum_{t=1}^{T} y_{\mathrm{ref},t}$ and $H = \sum_{t=1}^{T}\|y_{\mathrm{ref},t} - \bar{y}_{\mathrm{ref}}\|_2^2 > 0$ be the centered reference-output energy. The scale-aware quantity $\alpha_r = E\rho_r/H$ combines target energy and normalized fit reduction. Under the idealized rescaling $(d_t, e_t) \mapsto (\epsilon d_t, \epsilon e_t)$ and the implemented relative ridge $\lambda = \gamma\,\mathrm{tr}(G)/n$, $\gamma > 0$ and $\epsilon \neq 0$, the matrices G, C and λ scale by ϵ^2. Hence $B_\star$ is unchanged, $B_\star L$ scales by $|\epsilon|$, and $E\rho_r$ scales by ϵ^2. The ratio ρ_r is invariant, while α_r scales by ϵ^2 for a fixed reference energy H. Changing bit width is not generally a uniform rescaling, but this identity explains why a normalized spectrum cannot by itself identify large accuracy gains.

For a fitted correction $p_t = UVd_t + b$ on held-out tokens, define

$$\eta_{\mathrm{val}} = \frac{\sum_t \|e_t\|_2^2 - \sum_t \|e_t - p_t\|_2^2}{\sum_t \|y_{\mathrm{ref},t}\|_2^2}. \tag{15}$$

The denominator must be positive. Positive η_{val} means lower held-out layer error, with calibration factors and bias held fixed. A negative value detects local overcorrection. It is not a bound on language model loss: errors propagate through later nonlinear layers. This diagnostic uses validation inputs, separately from the fixed test windows used for PPL.

A.2 What the capacity controls identify

The rounded-input, unrounded-input and residual-input branches in Table 6 all have $r(n+m)+m$ scalar parameters and $r(n+m)$ factor MACs for each $m \times n$ layer. The unrounded-input branch is fitted to the same target, so the comparison isolates which input signal is supplied to an otherwise matched low-rank map.

A channel-restoration control with k selected input channels uses $mk + m$ parameters and mk MACs, excluding index storage and gather operations. Exact per-layer matching requires $k = r(n+m)/m$. The comparison below chooses k separately for square and rectangular projections so that its total parameter and factor-MAC budget equals that of rank-32 ResCom.

The strengthened Llama-3-8B control uses the same 128 training windows and seed zero as the original fit. For each layer, input-channel indices $j \in \{1, \ldots, n\}$ are ranked by $\sum_t d_{t,j}^2\|(W_s)_{:,j}\|_2^2$, including the stored SERQ compensator. The control restores $(W_s)_{:,I}d_I$ on the selected index set $I \subseteq \{1, \ldots, n\}$, $|I| = k$, and fits a mean bias. Query/output projections use $k = 64$ and key/value projections use $k = 160$. Both parameterizations use 27,590,656 scalar parameters and 27,262,976

factor MACs per token across the 128 repaired projections; the channel implementation additionally stores integer indices and gathers activations. The selected-channel and dense low-rank bases reach 7.1351 and 7.1465 PPL, respectively. Their similar performance shows that the benefit of the residual signal is not tied to one basis: the dense form provides a fixed-rank factorization, while the sparse form provides coordinate-level selection with indexed access.

B Post-calibration repair versus residual-aware weight splitting

ARHQ (Wang et al., 2026) and ResCom both preserve information unavailable from the rounded activation alone, but they solve different integration problems. ARHQ starts from a full-precision weight $W \in \mathbb{R}^{m\times n}$, chooses a rank-$r$ component $W_{\text{split}} = B_a A_a^\top$ with $A_a \in \mathbb{R}^{n\times r}$ and $B_a \in \mathbb{R}^{m\times r}$, and then quantizes the modified remainder. With weight quantizer Q_w and the shared bias omitted, its deployed layer is

$$\hat{y}_{\text{ARHQ}} = Q_w(W - W_{\text{split}})Q(x) + W_{\text{split}}x. \tag{16}$$

The high-precision branch is therefore coupled to a newly quantized main path.

ResCom instead starts from an already calibrated effective map W_s and keeps that map fixed. Its deployed layer is

$$\hat{y}_{\text{RC}} = W_s Q(x) + Bd + b, \qquad d = x - Q(x), \tag{17}$$

where $B = UV$ is fitted directly to the missing output contribution $e = W_s d$. This post-calibration objective allows the same repair procedure to attach to SERQ, PrefixQuant, or FlatQuant without repeating their weight calibration.

An algebraic expansion makes the distinction explicit. Adding and subtracting $Q_w(W)Q(x)$ in Eq. (16) gives

$$\hat{y}_{\text{ARHQ}} = Q_w(W)Q(x) + W_{\text{split}}d + \Delta_w Q(x), \qquad \Delta_w = Q_w(W - W_{\text{split}}) + W_{\text{split}} - Q_w(W). \tag{18}$$

The term $W_{\text{split}}d$ exposes a residual-dependent component, while $\Delta_w Q(x)$ records the accompanying change to the quantized main path. Because quantization is nonlinear, optimizing the split and requantizing the remainder generally changes both terms together. ResCom has no corresponding main-path change: it learns B from the frozen host layer's observed residual-to-output relation. Thus the two methods differ in optimization stage, branch input, fitted target, and whether deployment replaces the calibrated main operator; their shared access to unrounded information does not make their deployed corrections equivalent.

C Reproduction and implementation details

For completeness, the signal limitation used in the introduction can be stated directly. For two inputs $x_1, x_2 \in \mathbb{R}^n$ with $Q(x_1) = Q(x_2)$ but $W_s(x_1 - x_2) \neq 0$, any deterministic correction conditioned only on $Q(x)$ must be identical for both inputs, although their required corrections differ. Residual input distinguishes them. This is a statement about the information available to the branch; an unrounded-input branch also has that information and is tested as a matched-rank control.

The effective map in Eq. (1) includes the main quantized weight $\widehat{W} \in \mathbb{R}^{m\times n}$ and any frozen linear compensator. For the SERQ instantiation, $W_s = \widehat{W} + RS$, with $W_c = RS$, where $S \in \mathbb{R}^{k\times n}$ selects k salient channels and $R \in \mathbb{R}^{m\times k}$ is the stored compensator. This combined matrix is never materialized for inference. A different reference, when available, is written $y_{\text{ref}} = W_{\text{ref}}x + c$ with $W_{\text{ref}} \in \mathbb{R}^{m\times n}$; the output-error target remains $e = y_{\text{ref}} - \hat{y}$.

For the main SERQ experiment, the GPTQ checkpoint is loaded without another weight-quantization call. Its unmodified WikiText-2 perplexity is 7.7077 and its C4 perplexity is 11.1922. The original attention-only repair gives 7.1465 on WikiText-2. The fit reference includes SERQ's stored selected-column compensator. All repaired layers use phase-three asymmetric activation quantization in the evaluated configuration.

Uncentered moments and means are accumulated and then centered at solve time. In the SERQ runs, products are computed in FP32, accumulated into FP64 statistics, and solved in FP64 before

factors are cast to the loaded model dtype. The artifact-bound timing run confirms FP32 loading for the existing accuracy implementation. The 16-bit effective-width convention in Table 2 is therefore nominal storage accounting, not its in-memory runtime footprint. For Llama-3-8B, the attention-only branch contains 27,590,656 parameters including biases: 52.625 MiB at 16 bits and 105.25 MiB at 32 bits. Its factor MAC count is 0.3906% of decoder-block linear MACs; the all-linear variant is 1.2019%. The ratios are model-specific.

The original attention artifact required 252.8 seconds for collection and fitting on one B200, including 142.1 seconds of collection. Spectrum computation in subsequent diagnostic runs adds work. The cost is a forward calibration pass with dense moment accumulation and a factorization per layer, not merely an ordinary inference pass.

PrefixQuant and FlatQuant use their pre-rounding weight reference, including weight-side error; their target therefore differs from the activation-only SERQ target. PrefixQuant output-only rank-64 repair gives 7.6046. FlatQuant's 6.9600 uses output and down projections; its all-linear variant instead gives 6.9790. These distinct configurations are kept separate in the comparisons. The latter runs use FP32 moment accumulators.

D ACTIVATION-WIDTH RESULTS

The paired WikiText-2/C4 results across Llama and Mistral backbones are reported in the main-paper Table 3. The C4 documents are fixed across comparisons and unseen during WikiText-2 calibration.

D.1 WITHIN-BACKBONE WIDTH SWEEP

Table 9: Activation-width sweep on Llama-3-8B at fixed quantized weights. The sixteen-bit row is the same quantized host with activation quantization disabled, not the original full-precision model.

Activation width	Pipeline	RESCOM	change
4 bits	7.708	**7.147**	−7.28%
6 bits	6.552	6.545	−0.10%
8 bits	6.503	6.502	−0.01%
16 bits	6.499	–	0

The A16 control retains quantized weights; it is not the original FP16 model. For Llama-3-8B it gives 6.4991 perplexity. The rank-32 A4 repair recovers about 46.4% of the A16-to-A4 perplexity gap. This is a ratio of measured perplexities, not a layerwise error-energy fraction.

D.2 CROSS-BACKBONE ACTIVATION-WIDTH ANALYSIS

Table 10 compares the same checkpoints, repair rank, and attachment sites at four- and eight-bit activation precision.

Table 10: Activation-width sensitivity with SERQ. We retain the weights, repair rank, and attachment sites while changing the activation width. Each relative change is measured against SERQ at the same activation precision.

	Four-bit activations				Eight-bit activations		
Backbone	SERQ	RESCOM	change	ρ_{32}	SERQ	change	ρ_{32}
Llama-3-8B	7.708	7.147	−7.28%	0.294	6.503	−0.01%	0.289
LLaMA-2 7B	5.939	5.847	−1.55%	0.271	5.594	−0.01%	0.265
LLaMA-2 13B	5.227	5.148	−1.51%	0.270	4.977	−0.01%	0.261
LLaMA-3.2 3B	9.177	8.964	−2.32%	0.282	8.185	−0.02%	0.282
LLaMA-3.2 1B	12.461	12.071	−3.13%	0.355	10.411	−0.02%	0.356
Mistral-7B-v0.3	5.663	5.617	−0.81%	0.343	5.405	−0.01%	0.341

All six backbones in Table 10 have less than 0.1% relative perplexity change at A8, whereas A4 produces the larger paired reductions. The near-preserved ρ_{32} values show that normalized fit quality alone does not determine model-level gain: when activation rounding is already small, the absolute correction target is also small. The complete additional-family and generation measurements appear in Tables 16 and 17.

E ATTACHMENT SITES AND CALIBRATION SENSITIVITY

Table 11: Attachment-site and rank ablations on Llama-3-8B. Rank zero means a fitted bias only. Dashes denote unmeasured configurations. Site groups differ in parameter count, so they are not equal-budget comparisons.

Pipeline	Repaired layers	$r = 0$	$r = 16$	$r = 32$	$r = 64$
SERQ	all linear layers	–	–	**7.070**	–
SERQ	attention projections	7.715	7.167	7.147	7.128
SERQ	down projection	–	–	7.624	–
SERQ	output projection	–	–	7.672	–
SERQ	gate and up projections	–	–	7.688	–
PrefixQuant	output projection	7.790	–	7.646	**7.605**

The attention, down, and gate/up groups are fitted against the same unrepaired model, then loaded together. The combined Llama-3-8B artifact contains 224 branches and yields 7.0697 WikiText-2 and 10.2251 C4 perplexity, 58.16 MMLU, and 64.93 zero-shot average. Its larger parameter and MAC budget separates it from the default attention-only configuration.

E.1 CALIBRATION DRAWS AND SAMPLE COUNT

Table 12: Calibration-draw variability with test windows fixed. Positive Δ is host minus repaired perplexity. Standard deviations describe three or four draws as indicated by the final column; they are not test-set confidence intervals.

Backbone	Control	Repaired, mean	Δ mean	Δ s.d.	Helps on
Llama-3-8B	7.708	7.147	$+0.5604$	0.0059	4/4
LLaMA-2 7B	5.939	5.847	$+0.0916$	0.0030	3/3
Mistral-7B-v0.3	5.663	5.620	$+0.0431$	0.0024	4/4

Evaluation windows are fixed across calibration draws. The positive perplexity effects on Llama-3-8B, LLaMA-2 7B, and Mistral preserve their sign across the completed draws. These standard deviations measure fit variability, not paired downstream-task confidence intervals.

Table 13: Calibration sample-count sensitivity on Llama-3-8B. Rank 32, attention projections, seed zero; each control reproduces 7.7077.

2048-token windows	WikiText-2	C4	Mean ρ_{32}
32	7.1387	10.3471	0.2934
128	7.1465	10.3493	0.2942
512	7.1374	10.3522	0.2938

Large-model and additional-family completion. The LLaMA-2-70B comparison uses a separately verified GPTQ checkpoint with 128-channel weight groups and rank-32 corrections on its 320 attention projections. An independent single-writer reload reproduces WikiText-2 PPL 3.6378/3.5975 and C4 PPL 5.7031/5.6790 before/after correction. Both paths use FP16 model tensors; these are accuracy evaluations rather than timing comparisons with the smaller FP32 simulation models. The 70B panel in Table 2 reports the completed language-modeling, eight-task zero-shot,

and MMLU evaluations. Gemma-3-1B/4B likewise use complete paired evaluations for every metric reported in the paper.

F Downstream and generation evaluation

The eight-task suite includes PIQA for physical commonsense, SIQA for social reasoning, ARC-Easy/Challenge and OpenBookQA for science questions, and HellaSwag, WinoGrande, and BoolQ for contextual reasoning and reading comprehension (Bisk et al., 2020; Sap et al., 2019; Clark et al., 2018; Mihaylov et al., 2018; Zellers et al., 2019; Sakaguchi et al., 2020; Clark et al., 2019). The eight-task mean is calculated before rounding the task scores. Original and incremental task runs share raw PIQA scores and repair-layer counts as an overlap check. All 57 MMLU subjects are required; the question counts sum to 14,042. LAMBADA (Paperno et al., 2016) is reported separately: on Llama-3-8B, 71.92 for SERQ and 74.11 for attention repair.

The generation study uses paired SERQ/ResCom runs on two Llama and two Gemma backbones. Each LongBench task has all 200/200/500 examples respectively, with paired prompt hashes and answer lists matching exactly. The newly integrated Gemma results were independently reaggregated from 7,200 saved per-example LongBench metrics before rounding. Each GSM8K run covers 1,319 questions. Gemma-3-1B downstream measurements cover the complete eight tasks and all 57 MMLU subjects (14,042 questions). The Gemma-3-4B text-only extension is evaluated from its native multimodal checkpoint only after an output-parity check of the extracted text model.

GSM8K uses five shots and both strict exact-match and flexible-extraction filters from the evaluation harness. The completed Llama-3-8B W4A4 runs give $316/1319$ and $477/1319$ strict matches. Their individual standard errors are 1.18 and 1.32 percentage points. The exact difference of the unrounded rates is 12.21 points. LongBench uses its original benchmark, not LongBench-v2: QMSum and SAMSum use ROUGE-L F1 and RepoBench-P uses its official first-line code-similarity metric. The extension uses official prompts, greedy generation, 200/200/500 examples respectively, and a recorded 7,500-token input budget with middle truncation. The generation limits are 512/128/64 tokens. This input budget is an explicit implementation choice; SERQ's table alone does not specify every prompt/truncation detail needed for an exact match.

Table 14: Zero-shot task details for the principal W4A4 comparisons in Table 2. † marks values published by SERQ (Park et al., 2026); the unmarked SERQ and ResCom rows are our paired checkpoint evaluations under the common harness. Averages are formed before rounding.

Backbone	Method	ARC-c	ARC-e	BoolQ	HellaS.	OBQA	PIQA	SIQA	WinoG.	Avg.
LLaMA-2 7B	Full precision†	46.25	74.49	77.74	76.00	44.20	79.05	46.11	68.90	64.09
	SmoothQuant†	27.82	25.97	44.16	26.39	27.20	49.95	34.49	47.51	35.44
	SmoothQuant, g128†	39.85	65.40	68.59	67.99	37.80	74.48	41.45	61.64	57.15
	QuaRot†	43.00	71.59	72.84	72.30	41.60	77.20	33.06	64.64	59.53
	SpinQuant†	43.60	71.30	73.80	73.20	40.40	76.00	44.10	65.40	61.00
	SERQ	43.26	71.97	75.47	74.05	44.00	77.91	45.04	68.43	62.52
	ResCom	42.06	71.30	76.30	74.31	43.40	77.31	45.19	68.51	62.30
LLaMA-2 13B	Full precision†	49.23	77.53	80.61	79.38	45.20	80.52	47.39	72.38	66.53
	SmoothQuant†	27.73	26.81	38.47	25.72	25.80	49.24	34.70	47.75	34.53
	SmoothQuant, g128†	44.45	70.50	75.23	72.52	39.00	76.93	43.19	68.43	61.28
	QuaRot†	46.25	73.70	78.47	76.29	43.00	78.89	33.32	70.48	62.55
	SpinQuant†	49.00	76.30	78.20	77.10	42.80	79.30	46.30	69.50	64.80
	SERQ	49.40	76.14	80.95	78.62	46.00	79.49	46.93	71.11	66.08
	ResCom	49.66	76.22	81.44	78.41	45.20	79.65	46.72	72.45	66.22
Llama-3-8B	Full precision†	52.99	77.69	81.07	79.20	45.00	80.74	47.08	73.48	67.16
	SmoothQuant†	26.02	24.20	50.61	26.65	28.20	51.69	33.78	49.57	36.34
	SmoothQuant, g128†	30.38	46.59	61.83	55.61	33.60	64.20	39.92	59.59	48.97
	QuaRot†	43.26	69.57	70.49	72.22	42.80	77.04	32.96	64.64	59.12
	SpinQuant†	46.90	72.00	73.40	71.90	42.40	75.20	44.40	67.70	61.75
	SERQ	45.22	71.30	78.23	73.93	41.80	76.88	45.24	71.27	62.98
	ResCom	49.32	74.45	80.00	76.35	44.60	78.02	46.88	69.53	64.89

Table 15: Paired W4A4 zero-shot task details for the additional backbones in Tables 2 and 3. Both rows in each pair are our evaluations of the same checkpoint before and after repair.

Backbone	Method	ARC-c	ARC-e	BoolQ	HellaS.	OBQA	PIQA	SIQA	WinoG.	Avg.
Llama-3.2-1B	SERQ	34.30	53.32	60.06	58.35	34.60	69.53	41.25	56.04	50.93
	ResCom	33.11	55.22	59.45	58.84	35.20	69.59	40.84	55.88	51.02
Llama-3.2-3B	SERQ	40.10	67.13	70.70	69.29	37.40	75.63	45.45	66.14	58.98
	ResCom	42.32	67.42	71.22	70.78	40.20	76.77	45.91	66.46	60.14
Mistral-7B-v0.3	SERQ	51.71	78.96	80.92	78.39	46.60	80.85	46.06	73.32	67.10
	ResCom	52.65	78.96	80.83	79.36	45.40	81.12	46.32	72.45	67.13

Table 16: Additional-family W4A8 results. Each ResCom row uses the paired SERQ checkpoint and rank-32 attention repair. All task pairs are complete; bold compares each paired result.

Model	Bits	Method	WikiText-2	C4	0-shot	MMLU
Gemma-3-1B-PT	W4A8	SERQ	**15.026**	**19.104**	**55.81**	**25.13**
		ResCom	15.058	19.138	55.55	25.12
Gemma-3-4B-PT	W4A8	SERQ	**10.753**	**14.901**	65.20	**55.73**
		ResCom	11.128	15.326	**65.34**	55.52

Table 17: W4A8 generation results. †: published comparator values reported by SERQ (Park et al., 2026); SERQ rows are our paired checkpoint evaluations. Metrics and prompt settings match the W4A4 study in Table 5.

			GSM8K 5-shot ↑		LongBench ↑		
Model	Bits	Method	Flexible	Strict	QMSum	SAMSum	RepoBench-P
Llama-3-8B	W4A8	LLM.int4()†	7.88	4.62	13.29	34.20	44.93
		L^2QER†	36.24	35.33	7.28	25.81	47.03
	W4A8	SERQ	**43.52**	**43.14**	23.28	44.20	**65.98**
		ResCom	43.37	42.91	**23.53**	**44.45**	65.90
Llama-3.2-3B	W4A8	LLM.int4()†	6.76	6.44	15.83	36.42	35.24
		L^2QER†	18.95	18.42	21.53	40.63	60.18
	W4A8	SERQ	**20.17**	**20.09**	**23.15**	**43.95**	64.77
		ResCom	19.86	19.79	22.89	43.81	**64.94**
Gemma-3-1B-PT	W4A8	SERQ	**2.27**	**1.74**	17.94	7.16	**51.10**
		ResCom	2.12	1.59	**18.14**	**7.25**	50.96
Gemma-3-4B-PT	W4A8	SERQ	**28.81**	**28.58**	**21.62**	**8.54**	**65.34**
		ResCom	27.67	26.91	21.60	6.99	65.20

G IMPLEMENTATION AND END-TO-END MEASUREMENTS

Direct end-to-end comparison. Figure 5a uses direct runs of six Llama-3-8B configurations on one B200. Every timed model first completes the same 141 WikiText-2 windows. FP16 uses FP16 compute; the quantization-simulation models use FP32 with TF32 disabled. L^2QER retains both factor products and their intermediate/output quantizers at rank 64 on 224 linear layers. QuaRot uses its Hadamard rotation, GPTQ, and INT4 simulation. ResCom adds rank-32 factors on 128 attention projections to the SERQ checkpoint. Table 18 reports the underlying measurements.

Compact-storage protocol. Figure 5b uses a separate fresh-process measurement with 4096 input tokens and 32 generated tokens. Peak allocated memory includes codebooks, factors, KV cache, and workspace. Four-bit weight indices and FP32 per-group codebooks preserve the evaluated weight values; one layer is dequantized at a time for FP32 GEMM. Lossless weight round-trip and full-model logit-parity checks connect the packed representation to the evaluated checkpoint. Three repetitions give peak memory of 19.6862/14.1303/14.2330 GiB and mean wall time of 0.7150/3.6972/3.9455 s for FP16/SERQ/ResCom. Timing uses three groups, three warmups, and five prefill calls per group; cached decoding recreates the prefix cache and generates exactly 32 tokens.

Table 18: Numerical breakdown for Figure 5a. B200, Llama-3-8B, batch one, 4K prompt; milliseconds.

Method	Prefill	Time per output token
FP16	209.07	15.89
L^{2}QER-A8	2600.15	414.33
L^{2}QER-A4	2601.00	416.03
QuaRot	1470.75	78.70
SERQ	1342.32	76.79
ResCom	1372.46	86.11

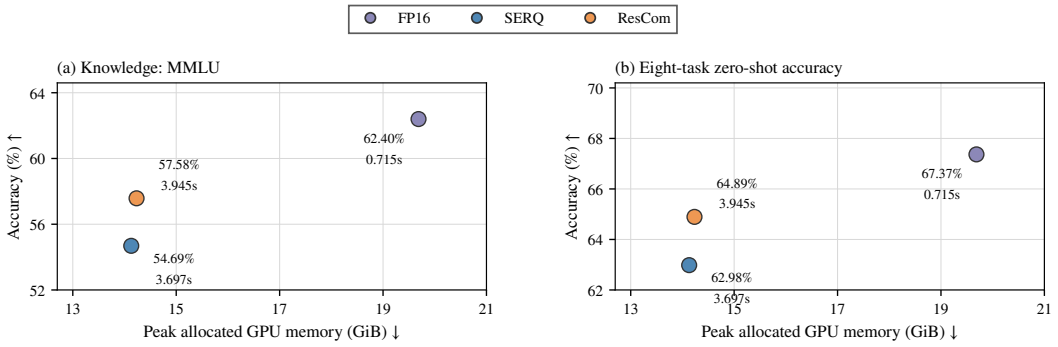


Figure 6: Full memory–accuracy views for MMLU and the eight-task mean, using the measurements summarized in Figure 5b. Each point is labeled with its measured end-to-end latency.

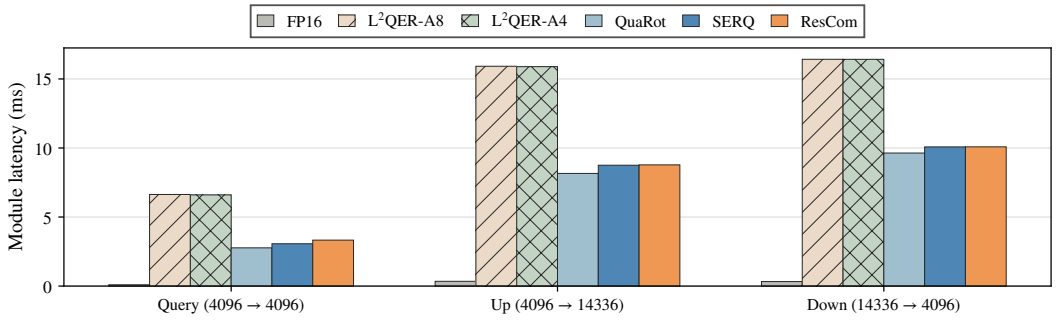


Figure 7: Per-module latency for the layer-0 query, up, and down projections of the six evaluated Llama-3-8B configurations on B200, batch one with 4096 input tokens. The measurements use each checkpoint's actual inputs and weights and complement the end-to-end results in Figure 5a.

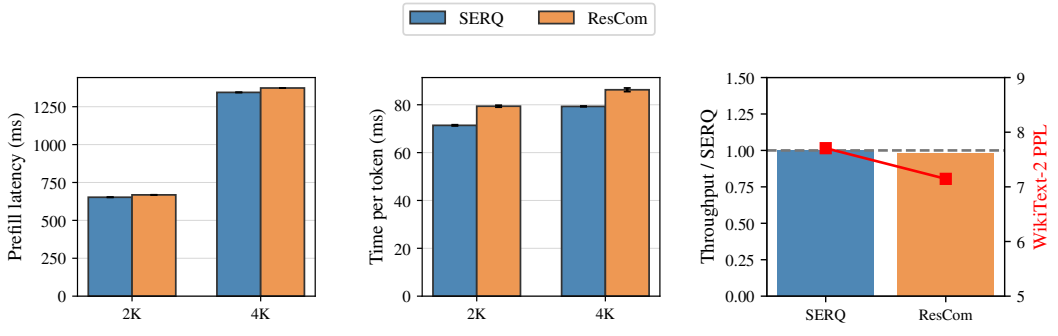


Figure 8: Paired SERQ and ResCom latency across prompt lengths on B200, batch one, with the corresponding WikiText-2 perplexity. Means and sample standard deviations use three interleaved groups. The shaded difference is the measured increment from the fitted rank-32 attention branch.

Fused residual frontend. The kernel in Table 19 uses the saved ResCom factors and captured validation activations. It fuses activation quantization with the first residual projection, emits two INT4 codes per byte with FP32 scale and zero point, and reduces group projections before the second factor product. All twelve shape/token tests reproduce the reference dequantized activations and keep relative correction error below 2×10^{-5}. Buffers are allocated before timing; each group uses ten warmups and fifty calls.

Table 19: Fused residual frontend with saved rank-32 factors on B200. Inputs are captured Llama-3-8B layer-0 activations; latency is in microseconds and averaged over three interleaved groups. The fused path also emits packed INT4 codes; main-path GEMM is outside this measurement.

Projection	Tokens	Eager	Fused	Ratio
q_proj	1	163.8	48.2	3.40
q_proj	16	167.9	52.1	3.22
q_proj	128	167.3	52.3	3.20
q_proj	2048	288.2	164.6	1.75
k_proj	1	160.2	46.8	3.42
k_proj	16	161.6	47.2	3.42
k_proj	128	168.8	51.3	3.29
k_proj	2048	282.3	155.6	1.81
o_proj	1	159.4	46.9	3.40
o_proj	16	167.9	51.2	3.28
o_proj	128	168.0	51.2	3.28
o_proj	2048	289.4	164.6	1.76

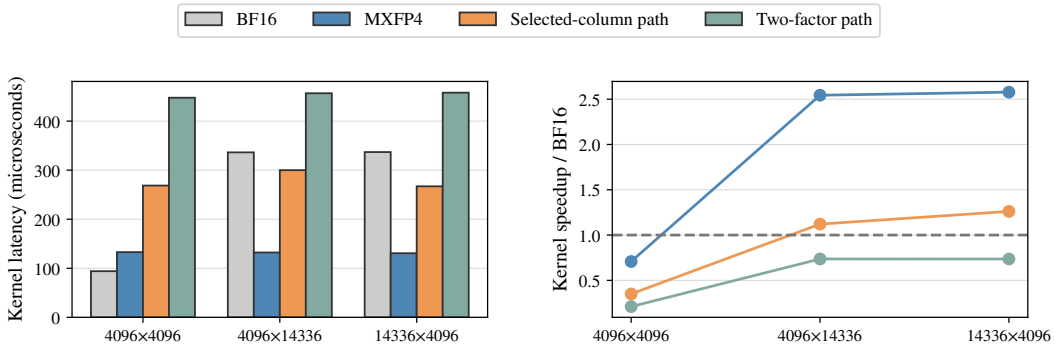


Figure 9: Packed kernel-component measurements for the selected-column path and the two-factor path. The former performs one thin product and the latter performs two. Shapes are (K, N), with input width K and output width N; BF16 and MXFP4 are the executed dtypes.

Native MXFP4 execution. A second implementation freshly quantizes Llama-3-8B and refits rank-32 ResCom factors for native MXFP4. All 224 linear projections invoke native MXFP4 matrix multiplication. Each fresh process runs batch one with the indicated prompt and exactly 32 generated tokens. Three repetitions include quantization, padding, residual computation, factor products, reconstruction, and KV-cache updates; checkpoint loading and fitting are excluded. Table 20 shows the resulting accuracy, end-to-end time, and peak allocation. This implementation verifies that the correction operates with a native low-bit main path; the current wrapper is a functional implementation rather than an optimized serving kernel.

Table 20: Llama-3-8B native MXFP4 execution on B200, batch one and 32 generated tokens. PPL is full WikiText-2 and shared across prompt lengths; time is the mean end-to-end seconds of three repetitions.

Prompt	Path	PPL ↓	Time (s) ↓	Peak (GiB) ↓
128	BF16	6.138	0.531	15.013
	MXFP4	7.865	6.826	8.172
	MXFP4 ResCom	7.733	7.017	8.275
2048	BF16	6.138	0.592	16.568
	MXFP4	7.865	6.854	9.588
	MXFP4 ResCom	7.733	7.186	9.690
4096	BF16	6.138	0.788	20.686
	MXFP4	7.865	7.302	11.097
	MXFP4 ResCom	7.733	7.571	11.200

H CALIBRATION TIME

Calibration collects moments and solves the rank-constrained regression once; task evaluation is excluded. Table 21 reports complete rank-32 attention-repair jobs. The independently timed Llama-3.2-3B run takes 184.4 seconds and reproduces the reported 8.9638 PPL, consistent with the 185-second rank-sweep job.

Table 21: One-time rank-32 attention-repair calibration on B200. Times include moment collection, fitting, and diagnostics; original PTQ and task evaluation are excluded.

Model	Llama-3.2 1B	Llama-3.2 3B	LLaMA-2 7B	LLaMA-2 13B
Fit time (s)	58	185	544	1353

All rows repair the query, key, value, and output projections. Fit time includes moment collection, factorization, and the diagnostic spectrum used in the paper. The corresponding control/repair PPL pairs are 9.1769/8.9638 for Llama-3.2-3B, 5.9389/5.8475 for LLaMA-2-7B, and 5.2274/5.1484 for LLaMA-2-13B.