
Calibration Leakage in a Layer-Wise Quantization Benchmark: With Fewer Calibration Rows Than Input Channels, In-Sample Scoring Understates GPTQ Error 2.6×

Yury Akinin (author) · Dmitrii Lozhkin (reviewer) · AVELIN AI · October 15, 2026

Abstract

Layer-wise quantization methods are often compared by output error on the calibration activations that also fit them. On all 192 MLP projection matrices of a 27B-parameter language model, encoded to NVFP4 by nine encoder settings, GPTQ at its default damping (0.01, raised to 0.1 for 189 of the 192 matrices) has a mean relative output error of 2.28% on its own 2,048 calibration rows, fewer than the layers have input channels, and 5.94% on 2,048 held-out rows, a factor of 2.6 (2.1 to 2.9 across three starting damping values); encoders that use no data move by at most 0.06 percentage points. GPTQ’s lead over a mean-squared-error scale search shrinks from 66% to 13%, but it still leads AWQ (6.38% with a full exponent grid, 6.23% with a local-Hessian search added). On the gate and up projections (rows at 40% of the input width) GPTQ wins on every matrix; on the down projection (12%) it trails an activation-weighted scale search at the default damping and leads it at damping 0.3, a value picked on the scoring rows. GPTQ’s published setting (rows at 51 and 15 times the input width) was not run. We recommend disjoint fit and score rows, and reporting the rows-to-width ratio with every layer-wise table.

1 Introduction

Post-training encoders for four-bit formats such as NVFP4 [6] choose scales and codes by a rule on the weights alone, by a search weighted with activation statistics [2], or, as in GPTQ [1], by rounding one column at a time and adjusting the remaining weights to cancel the error on calibration activations.

A common yardstick for choosing among them is local: the relative error of one layer’s output on calibration activations. For a method fitted to those activations it is a training loss, which GPTQ greedily minimizes. Layer-wise calibration is reported to resist overfitting unless the calibration set provides fewer equations than the layer has parameters [5]. What a practitioner needs is the size of the overfitting at LLM widths, where a small capture may hold fewer rows than input channels. GPTQ’s published setting, 128 segments of 2,048 tokens [1], gives 262,144 rows, about 51 times the 5,120 input channels of the gate and up projections studied here and 15 times the 17,408 of the down projection; calibration sets can be smaller, when data from one’s own domain is scarce or calibration time is short.

We measured it by accident. A table scored on the fit rows shows GPTQ ahead of the six non-GPTQ encoders by a factor of 2.6 to 3.4, and our first table was of this kind. A review of the benchmark code flagged the leak, and we reran the comparison on disjoint rows.

2 Setup

Model and format. The subject is a 27B-parameter decoder-only language model in BF16. Each of its 64 blocks has an MLP with gate and up projections of shape $17,408 \times 5,120$ and a down projection of shape $5,120 \times 17,408$: 192 matrices. All are encoded to NVFP4 [6]: 4-bit E2M1 codes, one 8-bit E4M3 scale per 16 consecutive input weights and one FP32 scale per tensor. All encoders share the rounding primitives of one codec, which reproduces a publicly released NVFP4 checkpoint shard (354 MB) byte for byte under the max rule. Only weight encoding is studied; activations stay unquantized.

Encoders. Three use no data. *Max* sets each block scale from the block’s largest magnitude; *MSE* tries 17 E4M3 scales around that value and keeps the one with the least squared weight error; *MSE+GS* also searches the global (per-tensor) scale over powers of $\sqrt{2}$. The others use calibration activations X , one row per token and d input channels. *AWQ* [2] rescales input channel j by $(a_j/a_{\max})^\alpha$, with a_j the mean absolute activation and α picked on the fit rows from a 22-point grid on $[0, 1]$ to minimize output error, then applies MSE; *AWQ+LH* (our label) picks α the same way and applies the Local-H search instead. *Local-H* (our label) runs the MSE search with each block’s error weighted by its 16×16 diagonal block of $X^\top X$; codes stay round-to-nearest. *GPTQ* [1] keeps the MSE scales and chooses the codes column by column with the error compensation of [3], using $H = 2X^\top X + \lambda \text{mean}(\text{diag } H) I$, 128-

column blocks and no activation ordering; λ starts at λ_0 and is multiplied by 10 until the Cholesky factorization succeeds. The default $\lambda_0 = 0.01$ ended at 0.1 for 189 of 192 matrices (0.01 for two, 1.0 for one); we also ran $\lambda_0 = 0.03$ (ended at 0.03 for 151 matrices, 0.3 for 41) and 0.3 (at 0.3 for all).

Data and metrics. Activations come from 16 chat conversations drawn from the model’s own training data, each cut to its last 2,048 tokens, in one pass through the unquantized model (blocks in fp32); the inputs of the MLP and of the down projection are captured at every block, so each matrix is judged on reference inputs and errors do not propagate. For an encoded matrix $\hat{W}$ we report

$$e_W = \|\hat{W} - W\|_F / \|W\|_F, \quad e_{\text{out}}(X) = \|(\hat{W} - W)X^T\|_F / \|WX^T\|_F. \quad (1)$$

Per matrix, 4,096 rows are sampled at a fixed stride across the conversations, in order; the first 2,048 fit and the last 2,048 score, so at most one conversation contributes to both. Each fit is scored on its own fit rows (in-sample) and on the held-out rows. With $n = 2,048$ fit rows, $n/d = 0.40$ for gate and up and 0.12 for down. Means are over the 192 matrices; 95% intervals are bootstrap intervals over the 64 blocks and do not cover calibration-draw variance. Encode times are from one V100S GPU.

3 Results

Table 1: Nine NVFP4 encoder settings, means over 192 matrices; each fit is scored on its own 2,048 fit rows and on 2,048 disjoint held-out rows. † Damping chosen on held-out rows, so optimistic. Bold: lowest weight and held-out error, † row excluded.

Encoder	Fit data	Weight error e_W , %	Output error e_{out} , %		Held-out / fit rows	Encode time, s per matrix
			on fit rows	on held-out rows		
Max	none	9.49	7.79	7.86	1.01	0.02
MSE	none	8.13	6.78	6.82	1.01	0.42
MSE+GS	none	8.12	6.78	6.82	1.01	2.97
AWQ	X	8.33	6.22	6.38	1.02	9.34
Local-H	X	8.45	6.03	6.29	1.04	0.64
AWQ+LH	X	8.44	5.97	6.23	1.04	13.08
GPTQ, $\lambda_0 = 0.01$ (default)	X	9.88	2.28	5.94	2.61	5.80
GPTQ, $\lambda_0 = 0.03$	X	10.60	2.19	6.30	2.87	5.74
GPTQ, $\lambda_0 = 0.3^\dagger$	X	9.08	2.67	5.62	2.11	5.71

The leak. The three data-free encoders are a control: they fit nothing, so any change comes from the score rows. They move by at most 0.06 percentage points (pp), the data-using encoders other than GPTQ (AWQ, Local-H, AWQ+LH) by at most 0.26 pp. GPTQ at the default damping moves from 2.28% to 5.94%, a factor of 2.61 (1.94 to 5.18 across matrices); its lead over MSE, 66% on the fit rows, is 13% on held-out rows. The factor shrinks as the applied damping grows (2.87 for $\lambda_0 = 0.03$, applied as 0.03 for 151 matrices; 2.61 at the default’s 0.1; 2.11 at 0.3) and does not vanish (Table 1, Figure 1a).

What survives. On held-out rows GPTQ at the default damping is still the best of the seven encoders, its damping variants aside (order: GPTQ, AWQ+LH, Local-H, AWQ, MSE+GS, MSE, Max): -0.88 pp against MSE (95% CI -0.98 to -0.78 ; lower on 175 of 192 matrices) and -0.35 pp against Local-H (-0.41 to -0.28 ; 157 of 192). Local-H recovers 61% of GPTQ’s gain over MSE at about one ninth of the encode time in our implementation. AWQ on the full grid has an interior optimum (0 of 192 matrices at either end; α from 0.40 to 0.85) and beats MSE by 6.5% on 192 of 192 matrices, yet stays 0.09 pp above Local-H. AWQ+LH is 0.06 pp below Local-H, 0.29 pp above GPTQ.

Damping. Held-out error falls with each increase of the applied damping (0.03, 0.1, 0.3) on every one of the 149 matrices that ended at those three values; mean fit-row error rises. At $\lambda_0 = 0.3$ GPTQ reaches 5.62%, the lowest held-out error in Table 1 (0.32 pp below the default). But 0.3 is the best of three values tried on the same held-out rows that score it, and the largest tried: the figure is optimistic and the optimum is not bracketed.

Where it does not. Table 2 splits the result by projection. At the default damping GPTQ beats MSE and Local-H on all 128 gate and up matrices ($n/d = 0.40$; at 0.3 too), but on the down projection ($n/d = 0.12$) its gain over MSE is 3.3% (47 of 64 blocks) and it is worse than Local-H by 0.17 pp (CI 0.07 to 0.27; better in 29 blocks). This depends on the damping: at 0.3 the down projection is 8.5% better than MSE and 2.9% better than Local-H (48 of 64 blocks).

Table 2: GPTQ by projection at the default damping and, for down, at $\lambda_0 = 0.3^\dagger$ ($n = 2,048$ fit rows; d is the input width). Differences are relative, negative favours GPTQ. Wins: matrices where GPTQ has the lower held-out error (64 per projection).

Projection	d	n/d	λ_0	GPTQ e_{out} , %			GPTQ vs. MSE, %			GPTQ vs. Local-H	
				fit rows	held-out	ratio	fit rows	held-out	wins	held-out, %	wins
gate, up	5,120	0.40	0.01	2.49	5.46	2.19	-62.4	-18.0	128/128	-10.0	128/128
down	17,408	0.12	0.01	1.85	6.91	3.73	-73.9	-3.3	47/64	+2.5	29/64
			$0.3^\dagger$	2.27	6.54	2.89	-68.0	-8.5	64/64	-2.9	48/64

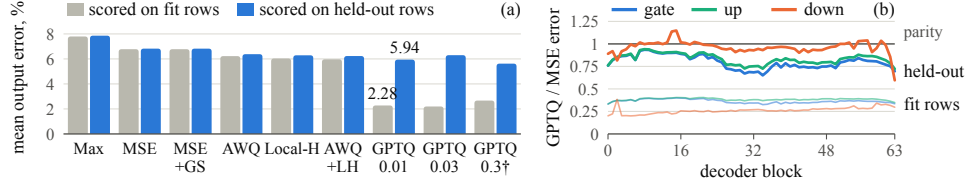


Figure 1: (a) Mean output error over 192 matrices on the fit rows and held-out rows of one fit; GPTQ groups give λ_0 . (b) GPTQ error relative to MSE at each block, default damping: on the fit rows (thin lines) the down projection looks like GPTQ’s largest gain, on held-out rows (thick lines) the smallest; GPTQ trails MSE in 17 of 64 down-projection blocks, in none for gate and up.

Why. With $n < d$ the fit rows span at most n of the d input directions. Error placed in the other $d - n$ directions costs nothing in sample, and GPTQ’s compensation moves rounding error there; fresh activations are not confined to that span. We did not measure the out-of-span energy, and n/d is confounded with projection type: this is an explanation, not an established law. **Weight error misleads.** The best encoder by it, MSE+GS (8.12%), is seventh by held-out output error, and the three GPTQ settings (0.01, 0.03, 0.3) rank 8, 9 and 6 of nine by weight error but 2, 5 and 1 by held-out output error (Table 1), in line with [4]: rounding to nearest is not optimal.

4 Limitations

This is one model, one in-distribution calibration pool and one value of n ; n/d is compared across projection types, not swept, and GPTQ’s published setting (n/d of 51 and 15) was not run. The metric is local: no accumulation over 64 blocks, no logit divergence or task accuracy. GPTQ ran with fixed scales and no activation ordering. The Hessian and e_{out} ran in half precision; fp32 reruns of AWQ and AWQ+LH (search and score, not the Hessian) moved e_{out} by at most 0.0002 pp. AWQ and AWQ+LH are our implementations, not compared with a public one, and omit AWQ’s clipping search (AWQ+LH is its nearest analogue); AWQ+LH took $\alpha = 0$, a grid edge, for 1 of 192 matrices.

5 Recommendations

1. Make fit and score rows disjoint by construction: two tensors, passed separately to the fitting and scoring code. Our leak came from independent row caps that all resolved to the same 2,048 rows.
2. Report n/d per module type with any layer-wise error; read a compensation tool’s layer loss as a training loss.
3. Keep data-free encoders in the table as a control; choose damping on rows apart from the scoring rows; rank by held-out output error, not by weight error; then confirm the finalists end to end.

Disclosure. Benchmark code and experiments were produced with AI coding agents under the author’s direction; the leak was first flagged by an automated adversarial code review. All numbers were recomputed from the per-matrix result files.

References

- [1] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *ICLR*, 2023. arXiv:2210.17323.
- [2] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *MLSys*, 2024. arXiv:2306.00978.
- [3] E. Frantar, S. P. Singh, and D. Alistarh. Optimal Brain Compression: A framework for accurate post-training quantization and pruning. In *NeurIPS*, 2022. arXiv:2208.11580.
- [4] M. Nagel, R. A. Amjad, M. van Baalen, C. Louizos, and T. Blankevoort. Up or down? Adaptive rounding for post-training quantization. In *ICML*, 2020. arXiv:2004.10568.
- [5] I. Hubara, Y. Nahshan, Y. Hanani, R. Banner, and D. Soudry. Accurate post training quantization with small calibration sets. In *ICML*, 2021 (PMLR 139:4466–4475). arXiv:2006.10518 (arXiv title differs).
- [6] E. Alvarez, O. Almog, E. Chung, et al. Introducing NVFP4 for efficient and accurate low-precision inference. NVIDIA Technical Blog, June 24, 2025.