MoE-VQ: A Model-Agnostic Gaussian Vector Codebook for 2-Bit Expert Weight Quantization

Anonymous authors
Paper under double-blind review

Abstract

Mixture-of-Experts models hold most of their parameters in sparsely activated experts, so memory rather than compute bounds deployment. At two bits a scalar grid loses more than any bit allocation recovers, and vector quantization recovers it only by fitting a codebook to the target model, which is expensive when a model has thousands of experts. We show that this cost can be avoided. Split each row into groups of 128 and divide by the group RMS, and expert weights are approximately Gaussian, and approximately so in the same way, across six MoE families from 14B to 250B parameters: skewness below 0.0035 in magnitude, excess kurtosis within $[-0.028, +0.170]$. One codebook of 2^{16} dimension-8 codewords, fitted once offline to $\mathcal{N}(0, I_8)$ and never shown a model, therefore serves every model, layer and expert. With block-LDLQ residual feedback and rank-14 compensation, MoE-VQ has the lowest perplexity of every configuration we ran that needs no inference-time rotation, on all four evaluated models, with each arm priced at its effective rate (asymmetric W4 is 4.25 bit/weight, not 4.0). It removes on average 52% of the excess perplexity over FP16 and gains 3.3–13.3 points of seven-task accuracy over tuned GPTQ, most of it on generative tasks.

1 Introduction

Mixture-of-Experts (MoE) architectures scale parameter count at fixed per-token compute and dominate recent large language models (Shazeer et al., 2017; Fedus et al., 2022; Jiang et al., 2024). The added capacity sits in sparsely activated expert weights, which account for 87.0–96.6% of the parameters of the four models studied here, so the deployment bottleneck is memory rather than compute: serving Mixtral-8x7B in FP16 requires 93.4 GB, i.e., two 80 GB H100s, where a single GPU suffices for the computation. Post-training quantization (PTQ) of expert weights is the most direct way to reduce this cost.

Most work on MoE quantization treats the problem as bit allocation: BitsMoE allocates bits per spectral component of a shared-basis SVD (Zhao et al., 2026), AlphaQ by layer sensitivity (Yang et al., 2026), MC-MoE and MxMoE by expert-level mixed precision (Huang et al., 2025b; Duanmu et al., 2025), and MoEQuant by correcting calibration imbalance (Hu et al., 2025). All of these decide how many bits go where, while the set of values a bit can represent (the grid) remains scalar. A scalar 2-bit grid, with four values per group, incurs a distortion that no allocation recovers: even a GPTQ baseline (Frantar et al., 2023) tuned with clipping search and sequential error compensation remains at $1.52\times$ FP16 perplexity on Mixtral-8x7B (Table 1). Vector quantization (VQ), which encodes several coordinates as one codeword, recovers the shaping gain that a scalar grid forfeits, and it leads 2-bit quantization of dense models in QuIP# (Tseng et al., 2024a), AQLM (Egiazarian et al., 2024) and QTIP (Tseng et al., 2024b); KBVQ-MoE applies it to experts after an SVD separates cross-expert redundancy (Xu et al., 2026). In all of these methods the codebook is built for the target model, which leads to a dichotomy: fixed codes such as the E8 lattice or a trellis carry no per-model cost but do not adapt to the distribution, whereas trained codebooks adapt but must be fitted for every model, a cost that is particularly high for MoE (Qwen3-30B-A3B alone has 6,144 experts).

Our starting point is the observation that this dichotomy can be avoided. If each row of an expert matrix is split into groups of $g = 128$ and RMS-normalized, the resulting distribution is close to

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.

a standard normal, with a shape that is nearly invariant across models, layers and experts: over six MoE architectures trained by different organizations on different data at different scales, skewness stays below 0.0035 in magnitude and excess kurtosis lies within $[-0.028, +0.170]$ (§2, Figure 1). Normalization fixes the variance and constrains the excess kurtosis only to $O(1/g)$, so this agreement in shape is a measurement rather than a construction (Appendix E). The same protocol extends beyond MoE: dense MLP weights share the shape and attention weights do not (Table 9), so the property belongs to *MLP-type* weights; leaving attention in FP16 is therefore a boundary rather than a convention, and MoE, where such weights dominate, is where the method is most useful. A codebook fitted once, offline, to $\mathcal{N}(0, I_8)$ therefore transfers to every model, layer and expert without retraining, a fair per-model refit matches it (Appendix D), and at 1 MiB it resides in L2 cache.

Building on this observation, we propose MoE-VQ (Figure 2). Its central component is a model-agnostic codebook of $K = 2^{16}$ codewords of dimension $v = 8$ fitted once to $\mathcal{N}(0, I_8)$, with an index cost of exactly 2.000 bit/weight. Two further components make it effective at two bits: block-LDLQ residual feedback, which generalizes GPTQ's sequential error compensation to 8-dimensional blocks, and rank-14 low-rank compensation, which requires no calibration data. Every comparison is reported at the effective bit rate of Eq. (2), including group scales and the low-rank term, for baselines as well (asymmetric W4A16 is 4.25 bpw, not 4.0). At two bits, MoE-VQ improves seven-task average accuracy over tuned GPTQ by 3.3–13.3 points on four MoE models and removes on average 52% of the excess perplexity over FP16 (Table 1); the weight memory of Mixtral-8x7B falls from 93.4 GB to 15.6 GB in the configuration whose quality we verify (13.6 GB with standard 4-bit attention; Table 23).

Our contributions are as follows.

- **A model-agnostic Gaussian codebook.** A 2^{16}-entry codebook fitted once to $\mathcal{N}(0, I_8)$ matches one fitted to a model's own weights (Appendix D), outperforms the E8 lattice by 1.0–3.2% at equal storage on all four models, and matches or outperforms the combination of QuIP#'s components without fine-tuning while requiring no inference-time rotation. The codebook size cannot be traded for rank either: at a matched rate, $K = 2^{12}$ with the freed bits spent on rank costs 16.9–35.4% in perplexity (Table 27).
- **MoE-VQ.** With block-LDLQ feedback and rank-14 compensation, the codebook improves seven-task accuracy over tuned GPTQ by 3.3–13.3 points at two bits, with the largest gains on generative tasks (Mixtral GSM8K $16.1 \rightarrow 44.9$; GPTQ solves 0/164 HumanEval problems, MoE-VQ 25.0%).
- **Routing preservation and value per bit.** Beyond the routing-change metrics of prior work (Chen et al., 2025; Park et al., 2026; Deng et al., 2026), we compare methods at a matched 2.000 bit/weight index rate: MoE-VQ preserves routing better than GPTQ, consistent with the gap widening more on generative than on likelihood tasks, and a gate-replay intervention bounds how much of the gap this channel can carry. Under effective-rate accounting the codebook's value per bit is stable across models, whereas that of low-rank compensation varies by 8.0× with matrix shape.

Related work is reviewed in Appendix C.

2 MOTIVATION

2.1 THE LIMIT OF SCALAR GRIDS

Quantized weights must be randomly accessible, so fixed-length codes are the practical constraint, and under this constraint the loss of scalar quantization is classical: the distortion-rate bound of a unit-variance Gaussian is 0.0625 at $R = 2$ (Cover & Thomas, 2006), while the optimal four-level scalar quantizer attains 0.1175, 1.88× above it (Max, 1960; Lloyd, 1982). The gap cannot be closed while coordinates are quantized one at a time: only joint coding makes cells rounder than hypercubes (space-filling gain) and places them according to the source density (shape gain) (Lookabaugh & Gray, 1989). Lattices recover only the former (Conway & Sloane, 1988); the shape gain requires a codebook trained on the distribution.

The same headroom is measurable in real models. Table 24 (Appendix K) varies only the coding dimension: the index rate is $\log_2 K/v = 2.000$ at every rung and the codebook carries no off-

set, so every rung stores one group scale and all three cost an identical 2.340 bit/weight under the same block-LDLQ feedback and rank-14 compensator. Excess perplexity over FP16 then decreases monotonically on all four models, from 34.0% at $v = 1$ to 20.9% at $v = 4$ and 16.8% at $v = 8$, and the reduction from v=1 to v=8 of 7.4–39.1 points is 11 to 115 times the replicate spread of the respective model.

The $v = 1$ rung separates the two gains: it is the Lloyd–Max quantizer for $\mathcal{N}(0, 1)$, hence shape-adapted where the tuned uniform baselines are not, yet it is the weaker quantizer, 34.0% against 24.4% for tuned uniform GPTQ with the same compensator, which spends *more* bits (2.47 bit/weight, since an asymmetric grid also stores a zero point; GPTQ's feedback is the one-dimensional case of LDLQ (Chee et al., 2023), so the pipelines are otherwise identical). At dimension one the clipping search and the zero point are worth more than the shape gain; the benefit of the codebook lies in the coding dimension. At dimension eight the codebook is also 1.0–3.2% better than the E8 lattice at identical storage on all four models, and 0.9–1.7% better when both are combined with a Hadamard rotation, so space filling alone is not sufficient either. Conversely, removing the feedback while keeping the codebook worsens perplexity by 2.85–11.76%; neither the grid nor the feedback substitutes for the other. What remains is the cost: on which distribution must a shape-adaptive codebook be trained, and must it be trained per model?

2.2 Architecture-agnostic Gaussianity of expert weights

The codebook must adapt to weights normalized by their group scale, since the quantizer of §3 encodes each row in groups of $g = 128$ divided by the group RMS. To test whether that distribution differs across models, layers and experts, we sampled expert projection matrices from six MoE architectures spanning 14B–250B parameters and compared 1.2M–4.0M normalized values per model with the standard normal (protocol in Appendix E).

Figure 1 shows the result. On a logarithmic scale the six empirical densities coincide with the standard normal across the central 99.9% of the mass (a); in the tails the empirical quantiles stay within 0.12 of the Gaussian quantiles (b); and skewness stays below 0.0035 in magnitude while excess kurtosis lies in $[-0.028, +0.170]$ (c; Table 7). The normalization accounts for almost none of this agreement: it fixes the variance and pins the excess kurtosis at $-6/(g+2) = -0.046$ (Appendix E), but leaves the quantiles, the skew and the tail weight free. Heavy tails, skew or multimodality could all have appeared, and in attention weights they do. Jointly, the eight coordinates a codeword quantizes are uncorrelated at the null level, and the dominant departure, a correlation of $-1/(g-1)$ between squared coordinates, is the one the normalization predicts (Table 8). The statistics are pooled per model, so they bound the aggregate rather than each layer or expert; the transfer probe of Appendix D is the evidence that bears on that question. Approximate normality of weights underlies the NF4 data type for dense models (Dettmers et al., 2023); we extend it to MoE experts and to invariance across architectures.

We also measured how far the shape extends. Under the same protocol (Table 9), expert and dense MLP weights are indistinguishable, with excess kurtosis in $[-0.030, +0.198]$ and $[+0.065, +0.118]$ respectively, whereas attention weights are not Gaussian: their excess kurtosis reaches $+4.863$, and the same codebook fits them up to 27% worse. The shape is therefore a property of ***MLP-type*** weights rather than of MoE: leaving attention in FP16 (§3.5) is the boundary of the codebook's premise rather than an inherited convention, and MoE, where MLP-type weights are 96.6% of the parameters, is where a method restricted to Gaussian-shaped weights covers nearly all parameters.

The implication is that the model can be removed from codebook training. If the target distribution is $\mathcal{N}(0, 1)$ regardless of the model, k-means run once on samples from $\mathcal{N}(0, I_8)$ yields a codebook (2^{16} entries, 1 MiB) that references no model and costs nothing per new one, satisfying the shape adaptation required by §2.1 and model-agnosticism at the same time. The cost of this idealization is small: at equal seed budgets a per-model codebook fitted to a model's own weights stays within $\pm 0.08\%$ of the synthetic one in held-out nearest-codeword distortion, with neither direction winning consistently (Tables 4 and 5), and through the whole pipeline the two are indistinguishable on two models, the synthetic codebook ahead on one corpus of one model, and the per-model fit ahead on Mixtral by 1.3–4.7% (Table 6).

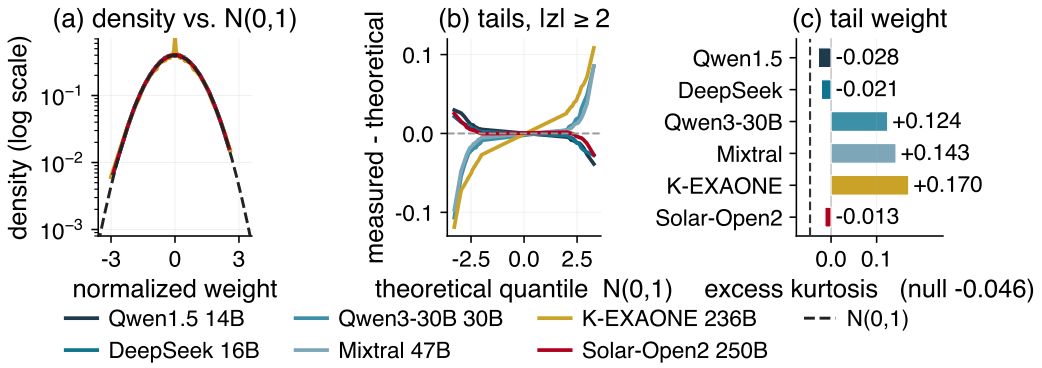


Figure 1: Group-normalized expert weights are approximately Gaussian, and closely aligned across architectures, in aggregate. Six MoE families (14B–250B), six tensors per model and 256 rows per tensor, RMS-normalized in groups of $g = 128$. (a) Density of each model's normalized weights (from 999 quantiles, 0.5% of the mass per bin) against the standard normal (dashed), log scale. (b) Empirical minus standard-normal quantile for $|z| \geq 2$. (c) Excess kurtosis per model against the null imposed by the normalization, $-6/(g+2) = -0.046$ (dashed; Appendix E).

3 Method: MoE-VQ

3.1 Notation and bit accounting

An MoE layer has E experts and a router that selects the top-k experts for each token and gate-weights their outputs. Each expert has a fused gate/up projection $W_{gu} \in \mathbb{R}^{2I \times H}$ and a down projection $W_{dn} \in \mathbb{R}^{H \times I}$, with I the expert inner dimension and H the hidden dimension; these two matrices are our target, and the router and attention keep the precision specified in §3.5. The layer-wise PTQ objective is the standard activation-aware reconstruction error

$$\min_{\hat{W}} \|(W - \hat{W})\, X^\top\|_F^2, \qquad H = X^\top X, \tag{1}$$

with X the calibration activations. Solving it greedily coordinate by coordinate while propagating the residual to the unquantized coordinates is GPTQ (Frantar et al., 2023), the one-dimensional case of LDLQ (Chee et al., 2023); §3.3 generalizes it to v-dimensional blocks.

Effective bit rate. Index bits alone are not comparable across quantizers, because group scales and low-rank terms must also be stored. We define the effective rate of one matrix as

$$\text{bpw} = \underbrace{\frac{\log_2 K}{v}}_{\text{index}} + \underbrace{\frac{16}{g}}_{\text{group scale}} + \underbrace{\frac{16\, r\, (O+I)}{O\, I}}_{\text{low-rank (BF16, rank } r)} \tag{2}$$

for a matrix of shape $O \times I$, codebook size K, codeword dimension v, group size g and rank r, and report it in every table. At the defaults $K = 2^{16}$, $v = 8$, $g = 128$, $r = 14$ the index term is exactly 2.000 bit/weight. The second term differs between methods. Our format carries no offset parameter: the codebook is fitted to a zero-mean source and the normalized weights are centred (§2.2), so a group stores one fp16 scale ($16/g = 0.125$). Asymmetric scalar quantization instead fits $[\min, \max]$ per group and therefore stores a scale *and* a zero point ($32/g = 0.25$); a 4-bit scalar baseline is therefore 4.25 bit/weight, not 4.0, and every comparison uses 4.25. The third term depends on the matrix shape; a model's rate is the parameter-weighted mean of its W_{gu} and W_{dn} rates, verified against packed bytes (Appendices A, G).

3.2 Model-agnostic Gaussian codebook

Following §2.2, the codebook $C \in \mathbb{R}^{K \times v}$ is trained once, offline, by Lloyd iterations on samples from $\mathcal{N}(0, I_8)$. It is initialized at the E8 lattice, so that a good spherical packing is the starting point and the iterations perform shape adaptation only, and fresh points are resampled at every iteration,

so that the Lloyd condition is optimized for the population rather than memorized (recipe and cost in Appendix D). The codebook consists of 2^{16} codewords of dimension eight, 1 MiB in BF16, which fits in an H100's 50 MB L2 cache but is eight times the size of an SM's L1 cache; this is what makes codeword lookup the dominant cost of the kernel (§4.3). The codebook is shared by every model, layer and expert. Encoding maps each block to its nearest codeword (tiled in units of 4,096 blocks; the tile size does not affect the results).

Group scale. One scalar per row group is initialized to the group RMS; after blockwise encoding at that scale, a single closed-form Lloyd update is applied,

$$s \leftarrow \frac{\sum_j \langle W_j, q_j \rangle}{\sum_j \langle q_j, q_j \rangle}, \tag{3}$$

with q_j the nearest codeword of W_j/s; this is the exact minimizer of the unweighted reconstruction error at fixed assignment.

3.3 BLOCK-LDLQ RESIDUAL FEEDBACK

The codebook alone ignores activation statistics, so we combine it with GPTQ's sequential error compensation generalized to v-dimensional blocks. With U the damped Cholesky inverse factor of H (1% of the mean diagonal added), blocks j are processed from left to right within each group:

$$Q_j \leftarrow s \cdot \mathrm{NN}(W_j/s), \tag{4}$$

$$E_j \leftarrow (W_j - Q_j)\, U_{jj}^{-1}, \tag{5}$$

$$W_{>j} \leftarrow W_{>j} - E_j\, U_{j,>j}, \tag{6}$$

propagating the residual to the remaining columns of the group and to all later groups. The codeword absorbs correlation within a block, so the feedback only has to handle correlation across blocks; the two are complementary. Removing the feedback while keeping the codebook worsens perplexity by 2.85–11.76% on all four models, 14–90 times their replicate spread (Table 1), and reverting the grid to scalar while keeping the feedback gives the ladder of §2.1.

3.4 LOW-RANK RESIDUAL COMPENSATION

The remaining weight error has a low-rank component, which we extract by randomized SVD and add back,

$$\hat{W} \leftarrow \hat{W} + U_r \Sigma_r V_r^\top, \qquad U \Sigma V^\top = \mathrm{SVD}_r(W - \hat{W}), \tag{7}$$

storing the factors in BF16 with $r = 14$. This stage sees only the weight-space error and needs no calibration data, in the manner of MiLo (Huang et al., 2025a). Its cost, the third term of Eq. (2), depends strongly on the matrix shape: 0.063–0.070 bit/weight on Mixtral, whose experts have a large inner dimension, against 0.255–0.401 on Qwen3-30B-A3B, whose experts are small and numerous. The same rank therefore costs very different numbers of bits, so rank must be judged jointly with rate; §4.4 shows that the shape dependence governs the utility per bit as well as the cost.

3.5 PROTOCOL

Expert weights are quantized to two bits; shared experts, which are active for every token, stay at four bits; the router stays in FP16 (ablated in §4.4). Calibration uses 128 sequences of 2,048 C4 tokens, layers are processed sequentially on activations from already quantized upstream layers, and evaluation uses a context of 2,048, the BitsMoE setting.

Each expert matrix is quantized against $H = X^\top X$ built only from the calibration tokens that the router actually sent to *that* expert (subsampled above 16,384 rows with a seed fixed per layer and expert), and the `down` projection is fitted against activations produced by the *already quantized* `gate_up`; the handling of rarely activated experts is described in Appendix B. Attention remains in FP16 in this protocol, and every quality number is measured in this setting; Table 23 reports the memory of that configuration alongside a deployment configuration with standard asymmetric W4 on attention and dense MLPs, whose saving is not a contribution of this method and whose quality we did not measure (§5). Quantization is simulated (weights are quantized and restored to BF16), so

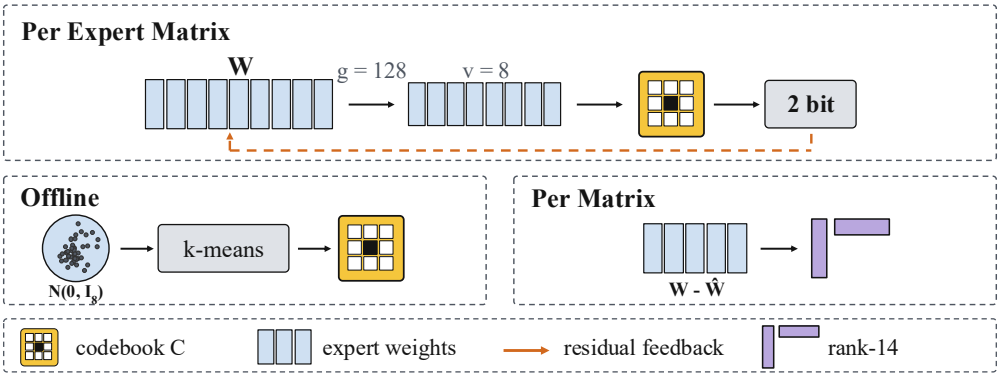


Figure 2: MoE-VQ in three stages. (i) A codebook is fitted once, offline, to $\mathcal{N}(0, I_8)$; it never sees a model and is shared by every model, layer and expert. (ii) Each expert matrix is normalized in row groups of g=128, cut into blocks of v=8 and mapped to its nearest codeword, with the block residual fed to the columns not yet quantized (block-LDLQ). (iii) The residual left by the two stages is absorbed by a rank-14 factorization.

quality reflects the quantization loss exactly; memory rests on Eq. (2) and on a packing format that is verified to be lossless and checked against a fused dequant+GEMV kernel on all eight production shapes (§4.3, Appendix G).

4 Experiments

4.1 Setup

Four MoE models form the main comparison: Qwen1.5-MoE-A2.7B (14B; 60 experts, one shared), DeepSeek-V2-Lite (16B; 64, two shared), Qwen3-30B-A3B-Base (30B; 128) and Mixtral-8x7B (47B; 8); expert counts span 8–128 and inner dimensions 768–14,336. Scaling is checked on 236B and 250B models (§4.4, Appendix H). Perplexity is measured on C4 at a context of 2,048, and accuracy follows the BitsMoE axis, four likelihood tasks and three generative ones. Every baseline is run by us under the same calibration set, sequential processing and bit accounting, with GPTQ tuned so that gains are not inflated by a weak baseline; the full list, the calibration and harness protocol, and the provenance of each row are given in Appendix B; our scorer agrees with the official lm-evaluation-harness within its standard error on seven of eight FP16 model–task cells (Table 20). The max–min range of perplexity over four independent runs of one configuration is 0.10% (Qwen1.5), 0.12% (DeepSeek), 0.34% (Mixtral) and 0.83% (Qwen3-30B) (Table 32); every judgment below is read against the spread of the respective model.

4.2 Main results

Table 1 compares all methods. Among the configurations that require no inference-time rotation, MoE-VQ has the lowest perplexity on all four models at two bits; our RHT variant lowers it further on two of them (Table 28). All arms were run by us under the same calibration and bit accounting; every arm indexes at 2.000 bit/weight and differs only in what it stores alongside the index, which the bit/weight column prices (2.25–2.47): tuned scalar quantizers, fixed-code variants (E8, with and without rotation) and dimension-4 VQ. AQLM, QTIP and KBVQ-MoE were not reproduced; KBVQ-MoE is compared on published numbers at its own rate in Appendix I: we lead on three of four models and trail on Mixtral in FP16-relative perplexity (Table 18), and retain more accuracy on all eight overlapping model–task cells (Table 19).

Four observations follow. First, the ceiling of scalar grids is sharp: on Mixtral, tuned GPTQ reaches 1.52× FP16 perplexity and AWQ 1.31×, and scalar variants with low-rank compensation or rotation reach 1.22× at best. Second, changing the grid crosses this ceiling: dimension-4 VQ alone beats the best scalar result on three of four models, and MoE-VQ lowers the excess perplexity over

Table 1: C4 perplexity (context 2,048) and Avg_4, the mean accuracy (%) over the four likelihood tasks, at two bits under the protocol of §3.5. bpw is the effective rate for DeepSeek-V2-Lite; per-model rates differ by at most 0.09 (Appendix A). The last two rows are an independent-batch replicate of MoE-VQ and its difference relative to the replicate spread. Bold values: best per column. Shaded rows: ours, with the default configuration in bold.

			Qwen1.5-MoE 14B		DeepSeek-V2-Lite 16B		Qwen3-30B-A3B		Mixtral-8x7B 47B	
Method	Grid	bpw	PPL↓	Avg_4 ↑	PPL↓	Avg_4 ↑	PPL↓	Avg_4 ↑	PPL↓	Avg_4 ↑
FP16	—	16.00	8.48	62.90	9.53	58.90	9.71	69.31	6.83	68.32
RTN	scalar	2.25	16.38	54.64	—	—	—	—	73.55	32.99
GPTQ (tuned)	scalar	2.25	11.19	57.91	12.84	52.16	12.06	64.72	10.37	56.90
AWQ	scalar	2.25	10.79	60.16	13.90	51.19	13.21	66.23	8.97	63.38
GPTQ + low-rank	scalar	2.47	9.97	59.55	12.58	52.18	11.44	66.21	8.89	63.42
AWQ + DRQ + low-rank	scalar	2.47	10.05	60.44	12.67	51.37	11.42	66.85	8.99	**65.48**
RHT + GPTQ	scalar	2.25	9.98	59.75	12.57	53.55	11.21	66.81	8.36	63.88
VQ dim-4 + low-rank	VQ	2.34	9.72	60.41	12.13	53.56	11.07	67.16	8.69	64.55
E8 lattice + low-rank	VQ	2.34	9.63	**61.29**	11.95	54.11	11.02	67.09	8.39	64.74
RHT + E8 lattice + low-rank	VQ	2.34	9.65	60.69	11.90	54.69	10.89	66.94	8.13	65.02
MoE-VQ w/o feedback (ours)	VQ	2.34	10.38	59.71	12.07	53.43	12.20	66.99	9.08	61.39
RHT + MoE-VQ (ours)	VQ	2.34	9.56	60.85	11.74	**54.88**	**10.77**	**67.95**	**7.99**	64.56
MoE-VQ (ours)	VQ	2.34	**9.54**	60.59	**11.73**	54.84	10.84	67.26	8.13	65.24
Batch check: MoE-VQ (ours), independent batch			9.53	60.69	11.74	54.47	10.94	67.57	8.12	65.44
difference / replicate spread			0.07 / 0.10 %		0.05 / 0.12 %		0.83 / 0.83 %		0.04 / 0.34 %	

FP16, averaged over the four models, from tuned GPTQ's 35.7% to 16.6%. Third, feedback and codebook do not substitute for each other: removing the feedback is worse on all four models and, on Qwen3-30B, worse than dimension-4 VQ. Fourth, a trained codebook beats an untrained lattice at equal storage: in the same pipeline our codebook is 1.0–3.2% better than the E8 lattice on all four models, and 0.9–1.7% better when both carry the Hadamard rotation. The RHT + E8 + LDLQ row reproduces QuIP#'s components without its fine-tuning; our method matches or beats it without a rotation at inference.

Table 2 breaks this down by task (plotted in Figure 3). The seven-task average improves by 3.3 (DeepSeek), 5.9 (Qwen3-30B), 6.2 (Qwen1.5) and 13.3 points (Mixtral). MoE-VQ is ahead in 27 of 28 model–task cells, and a paired exact test on the same items under the official harness resolves 20 of them, 19 in favour of MoE-VQ and one, MathQA on Qwen3-30B, against it (Table 21). Table 2 marks the unresolved cells. The structure of the resolved cells is informative: on the likelihood tasks the improvement reaches $+18.4$ points, on the generative tasks $+28.8$. On Mixtral, GSM8K rises from 16.07 to 44.88, and tuned GPTQ fails all 164 HumanEval problems, its generations degenerating into repetition, whereas MoE-VQ passes 25.0%; HumanEval on Qwen3-30B rises from 26.83 to 54.27. Re-measuring the same checkpoints with the official harness reproduces the HumanEval figures once two scoring conventions are accounted for (Table 22).

4.3 Memory and kernel

Table 23 (Appendix K) applies the bit accounting of §3.1 to each checkpoint in two configurations: (A), in which every quality number was measured (experts 2-bit, shared experts 4-bit, all else FP16), and (B), which additionally quantizes attention and dense MLPs with standard asymmetric W4. Under (A) the compression is 4.7–6.0× and Mixtral-8x7B drops from 93.4 GB to 15.6 GB, so its weights fit one H100 where FP16 needs two; on the large models the two configurations decide placement (§5, Appendix H). Every placement statement is weight-capacity accounting (parameter bytes only, not the KV cache, activations or workspace); serving at these placements is not demonstrated.

These are accounting predictions, so we checked them against real bytes with a fused dequant+GEMV kernel (Table 14, Appendix G): on all eight production shapes it matches the simulated quantizer to a relative error of at most 7.4×10^{-7}, and its packed-byte bit rate matches the shape-derived value to four decimals. At batch size one the reference kernel is slower than fp16, owing to a size-independent codeword-lookup cost (§5, Appendix G), which is an independent argument for QTIP's computed codes.

Table 2: Per-task accuracy (%) at two bits; the last three tasks are generative. Δ is MoE-VQ minus tuned GPTQ, and † marks $|\Delta|$ below 1.96 binomial standard errors (item counts in Appendix K). Bold: the better of the two quantizers.

Model	Method	HellaS.	MMLU	OpenB.	WinoG.	MathQA	GSM8K	HumanE.	Avg(7)
Qwen1.5-MoE 14B	FP16	77.71	61.02	44.20	68.67	35.75	60.12	33.54	54.43
	GPTQ (tuned)	71.64	52.68	40.00	67.32	30.59	34.65	14.63	44.50
	MoE-VQ (ours)	**75.03**	**57.72**	**40.40**	**69.22**	**32.80**	**52.01**	**27.44**	**50.66**
	Δ	+3.39	+5.04	+0.40†	+1.90†	+2.21†	+17.36	+12.81	+6.16
DeepSeek-V2-Lite 16B	FP16	74.78	49.66	44.00	67.17	34.47	23.28	18.90	44.61
	GPTQ (tuned)	65.82	36.51	40.40	65.90	29.11	8.49	6.71	36.13
	MoE-VQ (ours)	**68.44**	**42.39**	**42.00**	**66.54**	**30.62**	**14.03**	**12.20**	**39.46**
	Δ	+2.62	+5.88	+1.60†	+0.64†	+1.51†	+5.54	+5.49†	+3.33
Qwen3-30B-A3B 30B	FP16	81.39	78.67	44.40	72.77	60.57	89.84	70.73	71.20
	GPTQ (tuned)	76.38	69.43	41.40	71.67	52.53	73.39	26.83	58.80
	MoE-VQ (ours)	**77.93**	**75.42**	**42.20**	**73.48**	49.75	**80.14**	**54.27**	**64.74**
	Δ	+1.55	+5.99	+0.80†	+1.81†	−2.78	+6.75	+27.44	+5.94
Mixtral-8x7B 47B	FP16	82.76	67.65	47.00	75.85	36.58	61.49	34.76	58.01
	GPTQ (tuned)	74.10	44.92	42.20	66.38	29.45	16.07	0.00	39.02
	MoE-VQ (ours)	**78.70**	**63.30**	**43.80**	**75.14**	**35.34**	**44.88**	**25.00**	**52.31**
	Δ	+4.60	+18.38	+1.60†	+8.76	+5.89	+28.81	+25.00	+13.29

4.4 Scaling, bit allocation, and further studies

Across the six evaluated checkpoints we observe no monotonic increase in degradation with parameter count. The FP16-relative perplexity ratio lies between 1.084 and 1.230 from 14B to 250B, and the two large models average 1.134 against 1.165 for the four smaller ones (Table 25, Appendix K). Scale is confounded with expert count and width, top-k, router design, architecture and training data, so the observation is that degradation did not grow with scale here, not that it cannot (Appendix K.4). On the 236B model, MoE-VQ at two bits retains 91–98% of FP16 on instruction-following, knowledge and math tasks (Table 16), and a rerun on a different GPU architecture differs from the reported result by less than two runs on the same hardware do (Table 17).

Two further studies are reported in Appendix K. Varying one component at a time (Table 26, Figure 4) shows that the value of a bit is shape-dependent: the codebook axis is stable across models while the low-rank axis varies by $8.0\times$, since its cost term $16r(O+I)/(O\,I)$ spends far fewer bits on wide experts. Group size has its knee at $g = 128$, and a router-precision sweep prices full-precision routing. The comparison itself is robust: the ranking survives a change of perplexity corpus (Table 29), and the swings from changing the calibration sample (Table 30) or its domain (Table 31) stay below the gaps between methods, and are smaller for MoE-VQ than for GPTQ, as expected for a codebook that sees neither the model nor the corpus.

Routing preservation. We use the same top-k Jaccard as VSRAQ (Appendix C) and add two elements: methods are paired at a matched 2.000 bit/weight index rate, and the metric is connected to the likelihood/generative gap of §4.2. Paired in this way, MoE-VQ preserves routing better than tuned GPTQ on all four models, by 0.028 (DeepSeek) to 0.086 (Qwen1.5) in Jaccard (Table 10, Appendix F). The asymmetry follows: a likelihood task scores under teacher forcing, so a routing error perturbs a score without changing what the model reads next, whereas a generative task runs freely, so one token sent to the wrong expert changes the trajectory.

This evidence is correlational, so Appendix F bounds the channel by intervention (Table 11): replaying the FP16 gate outputs into the quantized model removes 11.1–34.4% of tuned GPTQ's damage (Table 13), MoE-VQ narrows this channel where it is large (Mixtral, 34.4% $\rightarrow$ 7.0%) and not measurably where it is small, and what the channel carries is how much each expert is weighted rather than which experts are chosen (Table 12). Because the replay imports more FP16 information than routing alone, this is an upper bound rather than a causal decomposition.

5 Limitations

The reference kernel is slower than fp16. At batch size one on an idle H100 it runs at $0.17\times$ fp16 on small expert matrices and $0.62\times$ on the largest. The cause is a size-independent lookup

cost: a 73× increase in parameters slows fp16 by 6.5× but our kernel by only 1.8×, and enlarging only the codebook from 2^{12} to 2^{16} adds 60.9% to the kernel time while the L2 gather bandwidth falls from 607 to 378 GB/s (Tables 14, 15). The memory claims are unaffected (the kernel's own bit rate matches Table 23 to four decimals on all eight shapes, and packing is lossless). Of the two remedies, a smaller K recovered with low-rank compensation is closed: it costs 16.9–35.4% in perplexity at a matched rate on all four models (Table 27, Appendix K.7), because rank saturates before it can compensate for a smaller table; computed codes in the manner of QTIP are the remaining route. QuIP, with its incoherence processing in PyTorch, reported about 1.5× the per-token latency of OPTQ (Chee et al., 2023). Batch sizes above one, prefill and end-to-end throughput were not measured; the memory figures are accounting predictions, not a serving demonstration.

No fine-tuning. QuIP# and AQLM finish with fine-tuning, and numbers that include this stage may exceed ours; AQLM and QTIP were not reproduced in our harness, and in Avg_4 the best cell of Table 1 belongs to a baseline on Qwen1.5 and Mixtral, while our RHT variant has it on DeepSeek and Qwen3-30B and our default configuration on none.

Scope, comparisons and anchors. Only expert weights are compressed (attention, activations and the KV cache are not), the same scope as BitsMoE; configuration (B) of Table 23 assumes standard W4 on attention and dense MLPs, whose quality we did not measure, and the 250B model fits on one card only under (B) while the 236B model fits under (A) (Appendix H). Model-agnosticism is a claim about the four models measured: on Mixtral a codebook fitted to that model does beat the shared one end to end (Appendix D), and we could not determine why. Stacking our codebook on allocation methods was not tried, and per-expert Hessians are built from each expert's own routed tokens without a minimum count (Appendix B), so calibration of rarely selected experts is untested. KBVQ-MoE has no public code, so it is compared on published numbers after FP16 normalization (Appendix I).

6 Conclusion

This paper began with a measurement: after group-scale normalization, MoE expert weights are approximately Gaussian, and approximately so in the same way, across six architectures from 14B to 250B parameters. If the distribution to which a codebook must adapt does not depend on the model, the codebook need not see the model, and 2^{16} codewords fitted once to $\mathcal{N}(0, I_8)$ serve every model, layer and expert without retraining. MoE-VQ combines this codebook with block-wise error feedback and low-rank compensation, none of which substitutes for another: at two bits it has the lowest perplexity of every configuration we ran that needs no inference-time rotation, on all four models, and it gains most on generative tasks, where it also preserves routing better. The reference kernel is still slower than fp16, for a cause traced to codeword lookup, and with the low-rank trade-off closed (§5) computed codes are the remaining route, under which the codebook would still never see the model.

AI use statement

In this work, we used generative AI tools for none of the tasks with required disclosure. We have not used generative AI tools for research ideation, core methodology design, data collection, or experimental analysis, and code generation or synthetic data tasks are not applicable to this work. Additionally, we used generative AI tools for grammar correction and language polishing. We have reviewed all AI-assisted work. Specifically, all drafts were initially authored by the researchers, and AI-suggested grammatical modifications were selectively adopted only after human verification to maintain technical clarity and accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

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 (NeurIPS)*, 2024. arXiv:2404.00456.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. QuIP: 2-Bit Quantization of Large Language Models With Guarantees. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2307.13304.

Yuanteng Chen, Yuantian Shao, Peisong Wang, and Jian Cheng. EAC-MoE: Expert-Selection Aware Compressor for Mixture-of-Experts Large Language Models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025. arXiv:2508.01625.

John H. Conway and Neil J. A. Sloane. *Sphere Packings, Lattices and Groups*. Springer-Verlag, New York, 1988. Grundlehren der mathematischen Wissenschaften 290.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2006.

Jianing Deng, Song Wang, Dongwei Wang, Zijie Liu, Tianlong Chen, Huanrui Yang, and Jingtong Hu. GEMQ: Global Expert-Level Mixed-Precision Quantization for MoE LLMs. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2605.23078.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient Finetuning of Quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.14314.

Haojie Duanmu, Xiuhong Li, Zhihang Yuan, Size Zheng, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. MxMoE: Mixed-precision Quantization for MoE with Accuracy and Performance Co-Design. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2505.05799.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme Compression of Large Language Models via Additive Quantization. In *International Conference on Machine Learning (ICML)*, 2024. arXiv:2401.06118.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. arXiv:2101.03961.

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 (ICLR)*, 2023. arXiv:2210.17323.

Parvel Gu. Detecting a Route Flip Is Easier Than Knowing Whether to Fix It: Causal Route-Mediated Damage in Quantized Mixture-of-Experts. *arXiv preprint arXiv:2608.11212*, 2026.

Xing Hu, Zhixuan Chen, Dawei Yang, Zukang Xu, Chen Xu, Zhihang Yuan, Sifan Zhou, and Jiangyong Yu. MoEQuant: Enhancing Quantization for Mixture-of-Experts Large Language Models via Expert-Balanced Sampling and Affinity Guidance. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2505.03804.

Beichen Huang, Yueming Yuan, Zelei Shao, and Minjia Zhang. MiLo: Efficient Quantized MoE Inference with Mixture of Low-Rank Compensators. In *Proceedings of Machine Learning and Systems (MLSys)*, 2025a. arXiv:2504.02658.

Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and Xiaojuan Qi. Mixture Compressor for Mixture-of-Experts LLMs Gains More. In *International Conference on Learning Representations (ICLR)*, 2025b. arXiv:2410.06270.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of Experts. *arXiv preprint arXiv:2401.04088*, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024. arXiv:2306.00978.

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 (ICLR)*, 2025. arXiv:2405.16406.

Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28 (2):129–137, 1982.

Tom D. Lookabaugh and Robert M. Gray. High-resolution quantization theory and the vector quantizer advantage. *IEEE Transactions on Information Theory*, 35(5):1020–1033, 1989.

Joel Max. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, 1960.

Hancheol Park, Geonho Lee, Tairen Piao, and Tae-Ho Kim. Value-and-Structure Alignment for Routing-Consistent Quantization of Mixture-of-Experts Models. *arXiv preprint arXiv:2606.05688*, 2026.

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 (ICLR)*, 2024. arXiv:2308.13137.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1701.06538.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. QuIP#: Even Better LLM Quantization with Hadamard Incoherence and Lattice Codebooks. In *International Conference on Machine Learning (ICML)*, 2024a. arXiv:2402.04396.

Albert Tseng, Qingyao Sun, David Hou, and Christopher De Sa. QTIP: Quantization with Trellises and Incoherence Processing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b. arXiv:2406.11235.

Zukang Xu, Zhixiong Zhao, Xing Hu, Zhixuan Chen, and Dawei Yang. KBVQ-MoE: KLT-guided SVD with Bias-Corrected Vector Quantization for MoE Large Language Models. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2602.11184.

Wanqi Yang, Yuexiao Ma, Alexander Conzelmann, Xiawu Zheng, Michael W. Mahoney, T. Konstantin Rusch, and Shiwei Liu. AlphaQ: Calibration-Free Bit Allocation for Mixture-of-Experts Quantization. In *ICLR 2026 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy (DeLTa)*, 2026. arXiv:2606.04980.

Jiayu Zhao, Zihan Teng, Minhao Fan, Tianrui Ma, Wentao Ren, Song Chen, and Weichen Liu. BitsMoE: Efficient Spectral Energy-Guided Bit Allocation for MoE LLM Quantization. *arXiv preprint arXiv:2606.00079*, 2026.

A Per-model effective bit rates

B Experimental setup

Models. Four MoE models form the main comparison: Qwen1.5-MoE-A2.7B (14B; 60 experts, one shared), DeepSeek-V2-Lite (16B; 64, two shared), Qwen3-30B-A3B-Base (30B; 128) and Mixtral-8x7B (47B; 8). Expert counts span 8–128 and inner dimensions 768–14,336, a wide spread of shapes. Scaling is checked on 236B and 250B models (§4.4, Appendix H).

Metrics and baselines. Perplexity is C4 at a context of 2,048. Accuracy follows the BitsMoE axis: four likelihood tasks (HellaSwag, MMLU, OpenBookQA, WinoGrande) and three generative tasks (MathQA, GSM8K, HumanEval). Every baseline is run by us under the same calibration set, sequential processing and bit accounting: RTN, GPTQ, AWQ, GPTQ and AWQ+DRQ with low-rank compensation, RHT+GPTQ, dimension-4 VQ, the E8 lattice with and without RHT, and our method without feedback. So that gains are not inflated by a weak baseline, GPTQ is tuned with clipping search, sequential quantization and domain-matched calibration ("tuned GPTQ"). Calibration draws 128 sequences from the C4 *train* shard and perplexity is measured on the C4 *validation* shard, so the two never share text. Our scoring reimplements lm-evaluation-harness v0.4 and agrees with the official harness 0.4.13, at the same `transformers` version, within the official standard error on seven of eight model–task cells (Appendix J). Every comparison is reported relative to FP16 measured in the same harness and the same batch; comparisons with published numbers are kept separate in Appendix I.

Per-expert Hessians. The row count of X differs from expert to expert by as much as the routing distribution does; above $16{,}384$ rows we subsample with a seed fixed per (layer, expert), and below it every routed token is used, with no minimum count. A rarely activated expert therefore receives a poorly conditioned or rank-deficient H, which is handled only by the damping (1% of the mean diagonal, raised by 10% up to three times if the Cholesky factorization fails), and the `down` projection inherits the error of the already quantized `gate_up` through its activations.

Replicate spread and provenance. The max–min range of perplexity over four independent runs of one configuration is 0.10% (Qwen1.5), 0.12% (DeepSeek), 0.34% (Mixtral) and 0.83% (Qwen3-30B) (Table 32); every judgment below is read against the spread of its model. Because replicate spread can exceed some method-to-method gaps, each row of Table 1 records its batch; our method run in two independent batches agrees within spread on all four models (last two rows), which licenses comparisons across rows.

C Related work

Weight quantization for LLMs. Post-training quantization has advanced largely by managing error on a scalar grid: GPTQ propagates each coordinate's residual to those not yet quantized (Frantar et al., 2023), AWQ rescales channels by activation magnitude (Lin et al., 2024), OmniQuant learns clipping and equivalent transforms (Shao et al., 2024), and QuaRot and SpinQuant rotate the coordinates (Ashkboos et al., 2024; Liu et al., 2025). The grid stays scalar, which fixes a loss at two bits that cannot be recovered (§2).

Vector and lattice quantization. For dense models, QuIP# combines an E8 lattice with Hadamard incoherence processing and LDLQ and finishes with fine-tuning (Tseng et al., 2024a); AQLM learns additive codebooks per model (Egiazarian et al., 2024); QTIP replaces lookup with computed trellis codes (Tseng et al., 2024b). Lattices do not adapt to the source and trained codebooks pay per model, a cost that is high for MoE; we remove the trade-off by fitting the codebook to a distribution rather than to a model. QuIP#'s three components without fine-tuning are a baseline in Table 1, which our method matches or beats without a rotation at inference (§4.2); we do not fine-tune, and methods that do may lead in absolute numbers (§5).

MoE quantization: allocation and format. Most MoE-specific work asks where bits should go: mixed precision by expert importance (Huang et al., 2025b; Duanmu et al., 2025), allocation over a shared-basis SVD (Zhao et al., 2026), calibration-free layer allocation (Yang et al., 2026), expert-balanced calibration (Hu et al., 2025), and global expert-level allocation (Deng et al., 2026). MiLo instead recovers low-bit loss with low-rank compensation (Huang et al., 2025a), the family to which our third stage belongs. All of these presuppose a scalar grid and are orthogonal to our method; stacking them is untested (§5). On the format axis, KBVQ-MoE applies dimension-4 VQ after a KLT-guided SVD computed from the target model (Xu et al., 2026), so its subspace is model-dependent where our codebook references no model (Appendices I, D); the two are combinable.

Routing-aware quantization. That quantization changes router decisions is a shared observation: EAC-MoE measures how often quantization changes the selected experts and reduces it by calibrat-

ing the router (Chen et al., 2025), VSRAQ uses the same top-k Jaccard as we do (Park et al., 2026), GEMQ and others define metrics of their own, and a diagnostic study finds route flips easy to detect but their harm unpredictable (Gu, 2026). Our contribution is not a metric but its use at a matched index rate, its connection to the likelihood/generative gap, and a gate-replay intervention that bounds what that channel can carry (§4.4).

Table 1 quotes the DeepSeek value of the effective bit rate for each row. The rate is shape-dependent through the low-rank term of Eq. (2), so Table 3 lists the per-matrix values behind every model. Three conventions matter. First, for routed experts the gate and up projections are quantized as one fused matrix of shape $(2I, H)$, so the low-rank term uses $O = 2I$; computing it per stored tensor instead gives 2.3935 for DeepSeek where the correct fused value is 2.3139. Second, a model's rate is the mean of its W_{gu} and W_{dn} rates weighted by parameter count: the fused matrix holds $2IH$ weights and the down projection IH, so $\mathrm{bpw} = (2\,b_{gu} + b_{dn})/3$. An unweighted mean overstates every model's rate by 0.001–0.024 because b_{dn} always exceeds b_{gu}. Third, the packed-byte accounting of Appendix G is the source of truth: the kernel recomputes each matrix's rate from its own bytes and the whole-model figure agrees with the weighted value here (Qwen3-30B: 2.4291 packed against 2.4288 derived).

Table 3: Effective bit rate of MoE-VQ per expert matrix at $K = 2^{16}$, $v = 8$, $g = 128$, $r = 14$; the model rate is the parameter-weighted mean.

Model	Matrix	Shape $O \times I$	Weights	Low-rank term	bpw	Model (weighted)
Qwen1.5-MoE / DeepSeek-V2-Lite	gate_up	2816×2048	$2IH$	0.1889	2.3139	2.3404
	down	2048×1408	IH	0.2685	2.3935	
Qwen3-30B-A3B	gate_up	1536×2048	$2IH$	0.2552	2.3802	2.4288
	down	2048×768	IH	0.4010	2.5260	
Mixtral-8x7B	gate_up	28672×4096	$2IH$	0.0625	2.1875	2.1901
	down	4096×14336	IH	0.0703	2.1953	
K-EXAONE-236B-A23B	gate_up	4096×6144	$2IH$	0.0911	2.2161	2.2344
	down	6144×2048	IH	0.1458	2.2708	

For the baselines, asymmetric scalar quantization stores a scale and a zero point per group, $32/g = 0.25$, so 2-bit scalar rows are 2.250 and W4A16 is 4.250 on every model; adding rank-14 compensation gives 2.4654 on DeepSeek, 2.5538 on Qwen3-30B and 2.3151 on Mixtral. The same rank therefore costs four to six times more bits on Qwen3-30B than on Mixtral, which is the shape dependence discussed in §3.4 and §4.4.

D CODEBOOK FITTING AND TRANSFERABILITY

Fitting. The codebook of §3.2 is fitted by 40 Lloyd iterations with $K = 2^{16}$, $v = 8$, initialized at the 56,881 E8 lattice points of squared norm at most 10 and filled to K with Gaussian samples. Each iteration draws 2×10^6 fresh samples from $\mathcal{N}(0, I_8)$, so the objective is the population Lloyd condition rather than an empirical one. Fitting takes minutes on one GPU and is done once for all models.

Does the codebook have to see the model? §2.2 argues from marginal moments that it does not. Moments are weak evidence for a joint codebook, so we test the claim where it can fail: we fit a codebook to a model's own weights under the same protocol and measure whether it improves the distortion on that model's held-out weights.

The fairness of this comparison rests on the sampling pool, so we specify it exactly. The real-weight pool is every held-in row (0–63) of every expert matrix of every layer and every expert, uniformly subsampled to 4.8×10^7 vectors (704–732 per centroid), with 2×10^6 fresh indices drawn per iteration, the same 40 iterations and the same E8 initialization as the synthetic fit. Held-out rows 64–127 are never read during fitting.

Pool size is the parameter that matters here, by enough to be worth recording for anyone repeating this experiment. A pool of 4×10^6 vectors taken in checkpoint order (early layers, and expert 0 only for stacked tensors) amounts to 61 vectors per centroid at $K = 2^{16}$, and per-model codebooks fitted in this way are 1.0–1.1% worse on held-out weights than the same fits at 704–732 per centroid (last

rows of Table 4). At this codebook size a per-model fit needs a pool an order of magnitude larger than the intuitive one before it represents the model rather than the sample.

Table 4: Nearest-codeword MSE on held-out rows 64–127 of every expert (2×10^6 vectors per model) and on fresh $\mathcal{N}(0, I_8)$. *Fair* fits use the full real-weight pool; the last two rows a 4×10^6-vector pool. Bold: the better of the synthetic codebook and the model's own fit.

Codebook	$\mathcal{N}(0, I_8)$	Qwen1.5	DeepSeek	Qwen3-30B	Mixtral
Synthetic $\mathcal{N}(0, I_8)$ (ours), best of 3 seeds	**0.08781**	**0.08719**	**0.08721**	0.08776	0.08776
Fair fit to Qwen1.5 weights	0.08796	0.08727	0.08730	0.08781	0.08784
Fair fit to DeepSeek weights	0.08796	0.08732	0.08729	0.08777	0.08781
Fair fit to Qwen3-30B weights	0.08804	0.08737	0.08737	**0.08773**	0.08771
Fair fit to Mixtral weights, best of 3 seeds	0.08806	0.08736	0.08738	0.08774	**0.08771**
Pool-size sensitivity: 61 vectors/centroid					
Fit to Qwen1.5 weights	0.08884	0.08822	0.08825	0.08872	0.08873
Fit to Mixtral weights	0.08893	0.08832	0.08833	0.08867	0.08865

The arms do not have equal seed budgets, so the table must be read at matched budgets. Three seeds were fitted for the synthetic codebook and for Mixtral, and one for each of the other three per-model fits; taking the *best* of three on the probe metric itself against a single-seed arm is a selection effect, so we compare like budgets: against a single-seed opponent we use our *worst* synthetic seed, and against Mixtral, which also has three seeds, best against best.

Table 5: Advantage (%) of the synthetic codebook over each model's own fit on that model's held-out weights (positive favours synthetic) under three seed selections, with the seed-to-seed spread as reference.

	Synthetic advantage (%)			Reference	
	best	mean	equal-budget	spread (%)	Verdict
Qwen1.5-MoE	$+0.096$	$+0.066$	$+0.046$	0.049	within spread
DeepSeek-V2-Lite	$+0.094$	$+0.085$	$+0.076$	0.018	**synthetic ahead**
Qwen3-30B-A3B	-0.034	-0.053	-0.077	0.043	**per-model ahead**
Mixtral-8x7B	-0.055	-0.067	-0.055	0.068	within spread

Two of the four verdicts change under this accounting, in opposite directions. On DeepSeek the synthetic codebook is better by 0.076% even at its worst seed against a 0.018% spread, so this advantage is real. On Qwen1.5 it is not: the worst-seed advantage of 0.046% falls inside the 0.049% spread, so the 0.10% that a best-of-three reading gives is an artifact of the selection. Symmetrically, on Qwen3-30B the per-model fit is ahead by 0.077% against a 0.043% spread, which the best-of-three reading had hidden as a 0.034% deficit "within spread". Mixtral, where both arms have three seeds, is within spread either way, though only marginally: comparing seed means rather than best seeds puts it at -0.067% against a 0.068% spread. The mean-against-mean reading changes none of the four verdicts.

The claim is therefore not that the synthetic codebook wins. It is that at equal seed budgets the two fits lie within $\pm 0.08\%$ of each other in held-out nearest-codeword distortion, the synthetic one ahead on one model, behind on one and tied on two. This symmetry is a cleaner statement of equivalence than a one-sided win would have been, and it is what the measurement supports. Because distortion is the quantity the codebook minimizes rather than the objective of the pipeline, in which block-LDLQ feedback, a Hessian-weighted target, the re-solved group scale and low-rank compensation act on top of it, we also ran the comparison end to end.

Through the full pipeline. Table 6 quantizes each model with both codebooks under the protocol of §3.5 and compares perplexity. One design choice determines whether the comparison is meaningful: the synthetic arm must be a *distribution*, not a single table. Four independently fitted synthetic codebooks span 0.23–0.41% of perplexity on these models, the same scale as the effect being measured, so a single-table comparison would report table-to-table variation as a codebook-family difference. We therefore fit the synthetic arm four times and credit the per-model codebook only when it lands outside that range.

Table 6: C4 and WikiText2 perplexity through the full pipeline with the synthetic codebook (range over four independently fitted tables) and a codebook fitted to each model's own weights. *outcome*: the better arm and its margin beyond the synthetic range (%); —: within the range.

	C4			WikiText2		
Model	synthetic (ours)	per-model	outcome	synthetic (ours)	per-model	outcome
Qwen1.5-MoE	9.524–9.547	9.565	synthetic -0.20	7.892–7.931	7.913	—
DeepSeek-V2-Lite	11.695–11.742	11.735	—	8.530–8.628	8.617	—
Qwen3-30B-A3B	10.852–10.878	10.863	—	7.125–7.169	7.125	—
Mixtral-8x7B	8.126–8.151	8.017	**per-model** -1.34	5.426–5.500	5.173	**per-model** -4.66

The outcome is again symmetric and again small. On Qwen3-30B and DeepSeek the per-model fit lands inside the synthetic range on both corpora, i.e., the two are indistinguishable. On Qwen1.5 the synthetic codebook is ahead by 0.195% on C4, but the same comparison on WikiText2 is a tie, so we do not count it. The one result that holds on both corpora is Mixtral, where the per-model codebook is better by 1.34% on C4 and 4.66% on WikiText2. End to end, therefore, fitting the codebook to the target model yields no systematic gain: the two codebooks are indistinguishable on two models, the synthetic one is ahead on one corpus of one model, and the per-model fit is ahead on one model, the only model here with eight wide experts rather than dozens of narrow ones. Both probes compare distribution-level k-means fits under an identical protocol, with rows 64–127 of every expert of every layer held out.

Position on the distortion ladder. At two bits and dimension eight the synthetic codebook's distortion on $\mathcal{N}(0, I_8)$ is 0.0878, below the 0.089 that QTIP reports for QuIP#'s E8P codebook at the same dimension (Tseng et al., 2024b); the optimal scalar quantizer is at 0.118 and the distortion-rate bound at 0.063. QTIP's trellis reaches 0.069 by coding in dimension 256, so it is lower than ours: our advantage over QTIP is not distortion but that no rotation is required at inference (Appendix C).

E GAUSSIANITY PROTOCOL AND MOMENTS

Sampling. For each of the six architectures we read expert projection tensors directly from the checkpoint shards, without loading the model: six tensors per model chosen uniformly across layers, alternating gate/up and down projections, and 256 rows per tensor. Shared experts and routers are excluded. Each row is divided into groups of $g = 128$ and divided by the group RMS, exactly as the quantizer of §3.2 normalizes before encoding. This yields 1.18M–4.00M normalized values per model. Quantiles are computed at 999 levels from 0.05% to 99.95%; Figure 1 reconstructs each model's density from them and compares the tail quantiles with those of the standard normal, the tail panel showing the levels with $|z| \geq 2$, i.e., below 2.3% and above 97.7%.

Table 7: Marginal moments of group-normalized expert weights; the variance is 1 by construction and the excess-kurtosis null is $-6/(g+2) = -0.046$ (Eq. (8)).

Model	n	Mean	Skewness	Excess kurtosis
Qwen1.5-MoE-A2.7B	2,162,688	0.0000	-0.0008	-0.0278
DeepSeek-V2-Lite	2,490,368	-0.0005	-0.0016	-0.0212
Qwen3-30B-A3B-Base	1,179,648	-0.0009	$+0.0010$	$+0.1243$
Mixtral-8x7B-v0.1	4,000,000	-0.0001	-0.0035	$+0.1426$
K-EXAONE-236B-A23B	4,000,000	-0.0006	-0.0012	$+0.1705$
Solar-Open2-250B	4,000,000	$+0.0002$	$+0.0008$	-0.0135

The null for excess kurtosis is $-6/(g+2)$, not 0. Dividing a group of g weights by their own RMS fixes $\sum_{i=1}^{g} x_i^2 = g$, so for a Gaussian source $x_i^2/g \sim \mathrm{Beta}(\frac{1}{2}, \frac{g-1}{2})$ and $\mathbb{E}[x_i^4] = g^2 \cdot 3/(g(g+2)) = 3g/(g+2)$. The excess kurtosis of exactly Gaussian data, *after* this normalization, is therefore

$$\frac{3g}{g+2} - 3 \;=\; -\frac{6}{g+2} \;=\; -0.0462 \qquad (g = 128), \tag{8}$$

exactly, not 0. Twelve independent samples of 3.2×10^7 group-normalized Gaussian values return -0.0459 ± 0.0003, confirming the derivation.

Read against Eq. (8), skewness, which the normalization leaves free, stays within ± 0.0035 across models trained by five organizations at scales from 14B to 250B, and excess kurtosis spans $[-0.028, +0.170]$, narrowing to $[-0.028, +0.143]$ over the four models used in the main text; these widths, 0.198 and 0.171, do not depend on where the null lies. All six models lie within $+0.018$ to $+0.217$ of the null, i.e., approximately Gaussian with slightly heavier tails.

The size of the constraint matters. Normalization accounts for only -0.046 of the shape, whereas attention weights in the same checkpoints, under the identical protocol, reach $+4.863$ (Table 9), two orders of magnitude more. The agreement in Table 7 is therefore a measurement about MLP-type weights, not an artifact of how they were normalized.

Joint structure. Marginal moments cannot see dependence between the eight coordinates a codeword quantizes, so we measured it directly on 2×10^6 held-out vectors per model (Table 8). Because every statistic below is nonzero at finite n even for exactly Gaussian data, each is also computed between two independent $\mathcal{N}(0, I_8)$ samples of the same size; that row is the null against which the model rows must be read.

Table 8: Joint diagnostics on 2×10^6 held-out 8-D vectors per model: correlation of coordinates (corr) and of squared coordinates (corr^2), energy distance and sliced Wasserstein-2 (512 directions) to $\mathcal{N}(0, I_8)$, and excess kurtosis. Null: two Gaussian samples of the same size (kurtosis from Eq. (8)).

	mean |corr|	max |corr|	mean |corr^2|	mean corr^2	energy	W_2^{sliced}	mean ex. kurt.
Null ($\mathcal{N}$ vs $\mathcal{N}$)	0.0006	0.0015	0.0005	-0.0001	4.7×10^{-5}	4.4×10^{-6}	-0.046
Qwen1.5-MoE-A2.7B	0.0008	0.0018	**0.0077**	$\mathbf{-0.0077}$	2.6×10^{-5}	2.2×10^{-5}	-0.022
DeepSeek-V2-Lite	0.0006	0.0018	**0.0081**	$\mathbf{-0.0081}$	4.8×10^{-5}	2.1×10^{-5}	-0.014
Qwen3-30B-A3B-Base	0.0006	0.0017	**0.0080**	$\mathbf{-0.0080}$	2.2×10^{-5}	1.8×10^{-5}	$+0.098$
Mixtral-8x7B-v0.1	0.0006	0.0014	**0.0080**	$\mathbf{-0.0080}$	3.1×10^{-5}	1.7×10^{-5}	$+0.101$

Linear dependence between coordinates is at the null level on all four models (mean |corr| 0.0006–0.0008 against a null of 0.0006). The squared coordinates are not: their mean correlation is -0.0077 to -0.0081, sixteen times the null magnitude and negative on every model. This value is predicted by the normalization. RMS normalization divides a group of $g = 128$ weights by their own root mean square, which fixes $\sum_{i=1}^{g} x_i^2 = g$ and therefore induces a correlation of exactly $-1/(g-1) = -0.0079$ between the squares of any two coordinates in the group. The measurement matches this prediction to within 0.0002. It is the same constraint that sets the marginal kurtosis null of Eq. (8), seen at second order across coordinates rather than at fourth order within one. The eight coordinates a codeword sees therefore lie on a thin shell, and $\mathcal{N}(0, I_8)$ idealizes the source by exactly this one, predicted constraint. Energy distance is at or below the null (the null is itself 4.7×10^{-5} at this n) and the sliced Wasserstein-2 distance is 4–5 times the null, small in absolute terms.

The premise of §3.2 is thus a group-normalized, hence sphered, approximately Gaussian population. Appendix D measures what this idealization costs: a codebook fitted to the real vectors, which carry the sphering constraint, is within $\pm 0.08\%$ of the synthetic one on held-out weights at equal seed budgets and indistinguishable end to end on three of four models.

How far the shape extends. Table 9 applies the same protocol to attention weights and to dense models. Expert and dense-MLP weights are indistinguishable; attention weights are not Gaussian, and the method therefore does not touch them.

DeepSeek-V2-Lite's attention is the mildest of the six at $+0.305$ excess kurtosis; it is the only model here using multi-head latent attention.

Cost. Reading six shards per model, normalizing, and computing quantiles and moments takes minutes per model on a CPU; the joint diagnostics take about nine minutes for all four models on one GPU. Both serve as diagnostics to run before applying the method to a new architecture.

Table 9: Skewness, excess kurtosis and nearest-codeword MSE under the shared codebook of group-normalized ($g = 128$) weights, by weight type; the two Llama-2 models are dense. Tensors are sampled in shard order, so values are comparable within this table only.

Model	Weights	Skewness	Excess kurtosis	Distortion
Qwen1.5-MoE-A2.7B	expert	-0.0007	-0.0296	0.087198
	attention	-0.0068	$+2.2017$	0.098195
DeepSeek-V2-Lite	expert	-0.0008	-0.0145	0.087183
	dense MLP	-0.0021	$+0.1184$	0.087894
	attention	-0.0015	$+0.3047$	0.088815
Qwen3-30B-A3B	expert	$+0.0003$	$+0.1980$	0.088370
	attention	-0.0022	$+1.2085$	0.093616
Mixtral-8x7B	expert	$+0.0011$	$+0.0382$	0.087499
	attention	-0.0041	$+3.2004$	0.101525
Llama-2-7B	dense MLP	$+0.0000$	$+0.0652$	0.087588
	attention	$+0.0025$	$+4.8626$	0.110662
Llama-2-70B	dense MLP	-0.0001	$+0.0661$	0.087615
	attention	$+0.0029$	$+2.5480$	0.099860

F ROUTING FACTORIAL DETAILS

Metrics. For a token batch passed through the FP16 model and through a quantized model, let T and T' be the top-k expert sets selected at one MoE layer for one token. Jaccard is $|T \cap T'|/|T \cup T'|$, the same definition as VSRAQ (Park et al., 2026); top-1 agreement is the fraction of tokens whose highest-scoring expert coincides; exact match is the fraction with $T = T'$. All three are averaged over tokens and MoE layers of the same batch, and the batch is identical across methods so that methods are paired. Router logits are not stored, so distribution-level divergences are not reported (§4.4).

Factorial design. Table 11 separates weight damage from routing damage in three passes over 16 C4 sequences of 2,048 tokens. Pass one runs the FP16 model and records, at every MoE layer, the gate outputs the router produces for each token (logits, routing weights and selected expert indices). Pass two runs the quantized model and measures its perplexity (*quant*). Pass three runs the quantized model again but replaces each layer's gate outputs with the recorded FP16 outputs (*oracle*), so that expert selection and gate weighting are those of FP16 while the expert weights are quantized. Then oracle $-$ FP16 is the damage attributable to weights alone and quant $-$ oracle the additional damage from changed routing. Because the recorded gate outputs are replayed as produced, the FP16 top-k selection and normalization are reproduced regardless of how a given architecture computes them, and the same code path serves all four models. One implementation detail matters: floating-point gate outputs are stored in fp16, but selected-expert indices are kept as integers, since casting an index tensor would corrupt the routing.

Separating which experts from how much. Replaying the entire gate output moves the selected experts and their gate weights together. Replacing one tuple element alone does not separate them: the weights the quantized model computed apply to *its* chosen experts, so substituting FP16's indices leaves a weight vector and an index vector describing different expert sets. This artifact is not small: on DeepSeek it made perplexity worse than no intervention at all (quant 9.824, index-only 12.868, weight-only 12.458). We therefore recompute the weights after substitution, using the model's own gate formula, which we determined empirically from the recorded tuples rather than assuming one per architecture: for DeepSeek and Qwen1.5 the weights are $\mathrm{softmax}$ over all logits gathered at the selected experts and left unnormalized (they sum to 0.34 and 0.22), while for Mixtral and Qwen3-30B the same quantity is renormalized to sum to one. Which of the two applies is detected at every call from the live tuple, and a mismatch raises an error rather than silently producing wrong weights. Table 12 reports the result.

The two halves are close to additive: their shares sum to 6.6–46.0% against 6.3–35.7% for the full replay, the excess being their interaction.

Table 10: Routing agreement with FP16 on the same token batch: Jaccard of the selected expert sets and top-1 agreement (Appendix F). The 236B rows compare different bit widths.

Model	Method	Jaccard↑	top-1↑
Qwen1.5-MoE-A2.7B	GPTQ 2-bit	0.766	0.834
	MoE-VQ 2-bit (ours)	**0.852**	**0.897**
DeepSeek-V2-Lite	GPTQ 2-bit	0.799	0.841
	MoE-VQ 2-bit (ours)	**0.827**	**0.866**
Qwen3-30B-A3B-Base	GPTQ 2-bit	0.790	0.734
	MoE-VQ 2-bit (ours)	**0.837**	**0.797**
Mixtral-8x7B-v0.1	GPTQ 2-bit	0.800	0.825
	MoE-VQ 2-bit (ours)	**0.872**	**0.893**
Not a paired comparison (bit widths differ)			
K-EXAONE-236B-A23B	GPTQ 4-bit	0.806	0.887
	MoE-VQ 2-bit (ours)	0.688	0.801

Batch-to-batch reproducibility. The decomposition was repeated on three disjoint batches of 16 sequences (offsets 0, 16 and 32 of the same C4 stream).

The main text quotes the batch means of Table 13.

Scope. The oracle uses gate outputs computed from FP16 hidden states and is not a deployable configuration; it is an upper bound that no router-correction technique can reach, and the route column of Table 11 shows only the room that bound leaves. Absolute perplexities in that table come from 16 sequences and differ from Table 1; the decomposition ratios are the object of interpretation. Routing for the 236B model (Table 10) is not a paired comparison because the available GPTQ baseline for that model is 4-bit.

G KERNEL DETAILS

Packed format. A quantized expert matrix is stored as 16-bit codeword indices (one per 8-dimensional block), one fp16 scale per row-group of 128, and the rank-14 factors in BF16. The shared codebook ($2^{16} \times 8$, 1 MiB in BF16) is stored once per deployment. Packing was verified to be a lossless re-encoding of the quantizer output: with fp32 scales and codebook the unpacked matrix equals the simulated-quantization matrix exactly (maximum absolute error zero); storing scales in fp16 introduces a relative error of 4×10^{-4}, an order of magnitude below the 2-bit quantization error itself.

Correctness and bit rate. The fused dequant+GEMV kernel was checked on the eight production expert shapes of Table 14 against a reference implementation, with relative errors between 1.6×10^{-7} and 7.4×10^{-7}. The kernel also computes the effective bit rate of each packed matrix from its own byte counts; on all eight shapes this agrees with the shape-derived values of Appendix A to four decimals, and the whole-model accounting from packing every expert tensor agrees with Table 23 (e.g., 2.4291 packed versus 2.4288 shape-derived for Qwen3-30B). These are the checks on which the memory numbers of §4.3 rest.

Throughput methodology. Throughput is measured on an idle H100 at batch size one for a single expert GEMV. The fp16 baseline is timed in the same process, interleaved with the VQ kernel, and both are reported as medians over 50 iterations after 10 warm-up iterations. This interleaving is necessary: timed in separate processes, fp16 on the same 4096×6144 shape gave 24.4, 62.5 and 63.0 μs across runs, a 2.6× spread that would make any reported speedup meaningless. The two kernel variants (v1 and v2) differ in load scheduling; v2 is 1.92× faster and is the one reported.

What the codebook-size ablation holds fixed. Table 15 changes only K on the 4096×6144 shape: the kernel, shape, group size and low-rank factors are identical, and the index width is 12, 14 or 16 bits. The L2 gather bandwidth reported there is the codebook-lookup traffic divided by

Table 11: Perplexity increase over FP16 on 16 C4 sequences of 2,048 tokens, split into *weight* (with FP16 gate outputs injected) and *route* (the remainder; share of the total in parentheses). Last column: share of our improvement over GPTQ due to the routing term.

		GPTQ (tuned)			MoE-VQ (ours)			Share of our gain
Model	FP16	total	weight	route	total	weight	route	due to routing
Qwen1.5-MoE 14B	7.33	+3.10	+2.22	+0.88 (28 %)	+1.18	+0.96	+0.22 (18 %)	34.5 %
DeepSeek-V2-Lite 16B	7.80	+3.38	+2.61	+0.77 (23 %)	+2.03	+1.50	+0.53 (26 %)	17.9 %
Qwen3-30B-A3B 30B	10.37	+2.34	+2.10	+0.25 (11 %)	+1.22	+1.09	+0.13 (11 %)	10.5 %
Mixtral-8x7B 47B	6.67	+3.77	+2.42	+1.34 (**36** %)	+1.47	+1.38	+0.09 (**6** %)	**54.6** %

kernel time; its fall from 607 to 378 GB/s as K grows is the signature of the lookup leaving cache. It is derived from traffic and time rather than from hardware counters, so it localizes the cost to the lookup without resolving which level of the hierarchy serves it.

Cost of the codebook. Fitting the shared codebook takes 51 s on one H100, once, for every model in this paper: 0.1% of the 12 h 32 m spent quantizing the 236B model and 3.4% of the fastest small-model run. Because the same 1 MiB artifact is reused across all six architectures, its amortized cost per model is smaller still, and a new architecture inherits it with no fitting or data-collection stage of its own.

H LARGE MODELS (236B, 250B)

The two large models are K-EXAONE-236B-A23B (236B parameters, 128 experts per layer) and Solar-Open2-250B (250B, 320 experts). Both were quantized with the protocol of §3.5 without any change to the codebook or hyperparameters. Their perplexity ratios and accuracy retention appear in Table 25: 20.37 → 24.13 (1.185) and 60.41 → 56.84% (94.1% retention) for the 236B model, and 9.35 → 10.13 (1.084) and 72.47 → 71.05% (98.0%) for the 250B model, the smallest degradation of the six models.

Task evaluations on the 236B model. Table 16 reports instruction following and knowledge tasks, including two Korean benchmarks, for FP16, standard W4A16 and MoE-VQ at two bits.

The 2-bit model retains 91–98% of FP16 on these tasks and sits within 0.1–5.1 points of the 4-bit scalar baseline. On IFEval the rate at which generations hit the length cap rises from 4.4% at FP16 to 7.8% at two bits, so part of the 3.1-point IFEval drop (0.8577 to 0.8262) may be truncation rather than quantization damage.

Cross-hardware reproduction. To test whether the 236B result depends on a particular hardware and software stack, we reran the pipeline end to end on a different GPU architecture; Table 17 reports the result. C4 perplexity is 24.28 against 24.13, or $+0.61\%$, and WikiText2 is 14.14 against 14.07, or $+0.49\%$. These differences require a reference scale. The main batches contain a second H100 run whose quantization configuration is identical and which differs only in the evaluations that followed it, and the two differ by $+0.71\%$ and $+0.66\%$. Changing the hardware, the CUDA and PyTorch build and the device map therefore moves the result less than running twice on the same hardware does. As a ratio to FP16 it is 1.192 against 1.185, so the value reported in Table 25 is not tied to particular hardware.

Memory and placement. From Table 23, the 236B model goes from 473.1 GB in FP16 to 79.8 GB in the quality-verified configuration (A), fitting a single 80 GB card with 5.7 GB to spare, and to 71.3 GB in the deployment configuration (B). The 250B model goes from 500.6 GB to 85.6 GB under (A), which exceeds the usable 85.5 GB by 0.13 GB and requires two cards; it fits one card only under (B), at 77.2 GB. FP16 serving of either model needs six cards by capacity (eight under the usual tensor-parallel convention). These figures follow from the bit accounting and the checkpoint tensor shapes.

Table 12: Routing channel split into expert selection and expert weighting (offset 0, 16 sequences). *full*: FP16 gate output replayed; *which experts*: FP16 selection with weights recomputed from the quantized logits; *how weighted*: quantized selection with weights recomputed from the FP16 logits. Shares are of the total damage, quant − FP16.

		Perplexity			Share of total damage		
Model	Method	full	which experts	how weighted	full	which	how
Qwen1.5-MoE 14B	GPTQ (tuned)	9.553	10.358	9.626	28.4 %	2.4 %	26.0 %
	MoE-VQ (ours)	8.291	8.467	8.316	18.4 %	3.4 %	16.3 %
DeepSeek-V2-Lite 16B	GPTQ (tuned)	10.404	11.081	10.492	22.9 %	2.8 %	20.2 %
	MoE-VQ (ours)	9.294	9.732	9.339	26.1 %	4.6 %	23.9 %
Qwen3-30B-A3B 30B	GPTQ (tuned)	12.466	12.592	12.504	10.6 %	5.2 %	9.0 %
	MoE-VQ (ours)	11.465	11.543	11.483	10.7 %	4.3 %	9.2 %
Mixtral-8x7B 47B	GPTQ (tuned)	9.086	9.193	9.935	35.7 %	32.9 %	13.2 %
	MoE-VQ (ours)	8.045	8.096	8.084	6.3 %	2.9 %	3.7 %

Routing. Table 10 lists the 236B model's routing agreement for reference only: the available GPTQ baseline for this model is 4-bit (Jaccard 0.806) and cannot be paired with our 2-bit result (0.688).

I Comparison with KBVQ-MoE on published numbers

KBVQ-MoE (Xu et al., 2026) is the closest prior work on the format axis: dimension-4 vector quantization of expert-specific components after a KLT-guided SVD separates the components shared across experts, plus a per-channel affine correction. It targets the same four models as this paper at two bits. Its repository contains only a README (115 bytes) and the camera-ready PDF, so the method was not reproduced; everything below uses the numbers published in the camera-ready.

Protocol differences. KBVQ-MoE evaluates WikiText2 at a context of 4,096 with RedPajama calibration, uses a dimension-4 codebook, and reports an effective rate of 2.08 bit/weight (dimension-4 indices plus a shared low-rank component and per-channel bias). We evaluate at 2,048 with C4 calibration and report 2.19–2.43 bit/weight (dimension-8 indices plus per-group scales and a per-matrix rank-14 term). Their seven tasks are ARC-e, ARC-c, HellaSwag, LAMBADA (OpenAI and standard), PIQA and WinoGrande; two overlap with ours.

Anchors. Because the two evaluations differ in context length and WikiText2 segmentation, absolute perplexities are not tabulated side by side; all comparisons below (Table 18) are relative to each paper's own FP16.

MoE-VQ has the lower FP16-relative perplexity on three of four models and KBVQ-MoE on Mixtral (1.389 versus 1.420).

Overlapping tasks. On the two tasks both papers report, FP16-relative retention (quantized accuracy divided by FP16 accuracy; their Tables 10–13, our Table 2) is given in Table 19:

MoE-VQ retains more in all eight cells. Retentions above 1.0 on WinoGrande are within noise for this task and should not be read as exceeding FP16; absolute accuracies are likewise not compared, since the anchors differ.

What distinguishes the two methods. KBVQ-MoE's KLT-SVD is computed from the target model's weights, so the subspace on which its codebook operates is model-dependent by construction (the paper describes it as a one-time KLT-SVD during calibration). Our codebook references no model, and Appendix D measures what that costs: at equal seed budgets a codebook fitted to a model's own weights lands within $\pm 0.08\%$ of the synthetic one on that model's held-out weights, ahead on one of four models and behind on one; through the whole pipeline the two are indistinguishable on three of four. The two designs are not exclusive; our codebook could be applied to their expert-specific components after their preprocessing, which we list as future work.

Table 13: Routing decomposition over three disjoint batches of 16 C4 sequences (offsets 0, 16, 32). Left: share of each method's damage removed by the full gate replay. Right: share of MoE-VQ's improvement over GPTQ due to the routing term.

	Routing share of damage (%)				Routing share of our gain (%)			
Model	GPTQ	MoE-VQ (ours)	mean Δ	range	o0	o16	o32	mean
Qwen1.5-MoE 14B	27.2	15.6	-11.6	5.3	34.5	32.3	35.7	34.2
DeepSeek-V2-Lite 16B	16.1	20.3	+4.2	11.1	17.9	8.4	-2.1	8.1
Qwen3-30B-A3B 30B	11.1	11.5	+0.5	1.9	10.5	8.7	12.6	10.6
Mixtral-8x7B 47B	34.4	7.0	-27.4	3.4	54.6	50.9	48.8	51.4

J HARNESS VERIFICATION

Setup. Our evaluation reimplements the scoring formulations of lm-evaluation-harness v0.4 so that quantized checkpoints can be scored in the same process that quantizes them. To check this reimplementation we ran the official harness, version 0.4.13, on two FP16 models, DeepSeek-V2-Lite and Qwen3-30B-A3B-Base. The comparison is designed to vary only the harness: `transformers` is pinned to 5.14.1, the version used for every result in this paper, and `trust_remote_code` is off, as in our own loader, so both harnesses run the same model implementation. Tasks, precision (bfloat16) and the zero-shot setting match our protocol. Two practical notes: lm-eval 0.4.9 cannot be used here, because it imports symbols removed in `transformers` 5 and because its task configs use unnamespaced dataset ids that the current `huggingface_hub` rejects; and DeepSeek's remote modeling code imports `is_torch_fx_available`, also removed in `transformers` 5, so `trust_remote_code` must be off rather than a choice of convenience.

What it shows. Seven of the eight model–task cells of Table 20 agree within the official harness's own standard error, and five agree to within a quarter of it (HellaSwag, MMLU, OpenBookQA and WinoGrande on Qwen3-30B, and WinoGrande on DeepSeek): MMLU differs by 0.15 points on DeepSeek and 0.01 on Qwen3-30B, WinoGrande by 0.15 and 0.24, and HellaSwag by 0.03 on Qwen3-30B. The remaining cell, HellaSwag on DeepSeek, differs by 1.35 points ($3.1\times$ the standard error).

The quantized checkpoints, seven tasks, and a paired test. The check above covers FP16 and the four likelihood tasks. We then ran the official harness on the eight saved 2-bit checkpoints themselves (four models $\times$ {tuned GPTQ, MoE-VQ}) across all seven tasks, logging per-item outcomes. Table 21 reports the result. Because the same items are scored for both methods and the outcomes are now stored, the comparison is a *paired* exact McNemar test rather than the unpaired binomial-SE comparison of Table 2. Pairing is the stronger test and it resolves cells the unpaired one could not: OpenBookQA on Mixtral, which Table 2 marks as noise, is $p = 3.5 \times 10^{-10}$ paired (12 items won by GPTQ against 66 by MoE-VQ). Of the 28 cells, 20 separate at $p < 0.05$; 19 favour MoE-VQ and one favours GPTQ, MathQA on Qwen3-30B, the same single loss that Table 2 reports.

HumanEval scoring conventions. On HumanEval the two harnesses differ by scoring convention rather than by model behaviour. Table 22 shows this by re-scoring the *official harness's own generations* under successively more permissive conventions.

The first cause is a tokenizer artifact specific to Mixtral. Decoding the completion tokens alone drops the leading `U+2581` word-start marker that SentencePiece attaches to the first token, so 163 of Mixtral's 164 completions come back with three leading spaces instead of four; concatenating prompt and completion then raises `IndentationError` and every solution fails regardless of its logic. Our generation path decodes the whole sequence and slices the prompt off the decoded string, which preserves the space. Restoring it moves Mixtral from 0.61 to 24.39. Exactly one line is short, so the repair must pad only the first: padding every line shifts a statement that belongs at the function's own level one column deeper and the code still does not run. This artifact should be checked for whenever a SentencePiece model is benchmarked on HumanEval: a scorer that does not normalise the leading whitespace will report approximately zero.

Table 14: Fused dequant+GEMV kernel on the eight production shapes. Top: relative error against the reference implementation and the bit rate computed from the packed bytes. Bottom: batch-1 latency on an idle H100 (medians; fp16 timed interleaved in the same process).

Shape	Role	Params	Rel. error	Kernel bpw	Table 23 bpw
2048×768	Qwen3-30B down	1.6 M	1.6×10^{-7}	2.5260	2.5260
2048×1408	DS/Qwen1.5 down	2.9 M	2.3×10^{-7}	2.3935	2.3935
1536×2048	Qwen3-30B gate_up	3.1 M	3.2×10^{-7}	2.3802	2.3802
2816×2048	DS/Qwen1.5 gate_up	5.8 M	2.4×10^{-7}	2.3139	2.3139
6144×2048	K-EXAONE down	12.6 M	3.7×10^{-7}	2.2708	2.2708
4096×6144	K-EXAONE gate_up	25.2 M	3.3×10^{-7}	2.2161	2.2161
4096×14336	Mixtral down	58.7 M	7.4×10^{-7}	2.1953	2.1953
28672×4096	Mixtral gate_up	117.4 M	3.7×10^{-7}	2.1875	2.1875
Throughput (batch 1)			MoE-VQ (ours)	fp16	speedup
2048×768	Qwen3-30B down	1.6 M	76.5 μs	12.8 μs	$0.167\times$
2048×1408	DS/Qwen1.5 down	2.9 M	76.5 μs	12.7 μs	$0.166\times$
1536×2048	Qwen3-30B gate_up	3.1 M	76.9 μs	12.9 μs	$0.167\times$
2816×2048	DS/Qwen1.5 gate_up	5.8 M	76.9 μs	12.8 μs	$0.166\times$
6144×2048	K-EXAONE down	12.6 M	78.0 μs	21.4 μs	$0.275\times$
4096×6144	K-EXAONE gate_up	25.2 M	110.2 μs	62.1 μs	$0.564\times$
4096×14336	Mixtral down	58.7 M	91.7 μs	47.4 μs	$0.517\times$
28672×4096	Mixtral gate_up	117.4 M	135.2 μs	83.5 μs	$\mathbf{0.618}\times$

The second is a scoring convention. Our scorer treats a markdown fence as a stop sequence and the official harness's `until` list does not, so a fenced solution reaches the interpreter with ``` still attached and raises `SyntaxError`. This costs Qwen3-30B 22 of 164 items and 13.4 points. Neither behaviour is a bug (bigcode-evaluation-harness strips fences, lm-evaluation-harness scores the text literally), but the choice is worth stating, because it is the larger half of the Qwen3-30B gap and it favours both methods (tuned GPTQ 24.39 to 28.66 as well).

What remains after both corrections is one item on Mixtral (24.39 against 25.00) and three on Qwen3-30B (52.44 against 54.27). Greedy decoding is not bitwise reproducible across batching schemes: re-running our own path on Qwen3-30B reproduces 54.27 exactly, yet only 84 of its 164 completions are byte-identical to the official harness's, and the ones that differ do so in substance (one returns `i % 2 == 0` where the other returns `i % 2 == 1`). Five of those flip pass or fail, netting three. The residual is therefore the scale of this variation, not a systematic gap, and Table 2's HumanEval column stands as an independent harness measures it.

K Additional results

K.1 Per-task accuracy

Figure 3 plots Table 2. Three details behind that table are recorded here rather than in the main text. *The dagger test.* Item counts are 10,042 (HellaSwag), 14,042 (MMLU), 500 (OpenBookQA), 1,267 (WinoGrande), 2,985 (MathQA), 1,319 (GSM8K) and 164 (HumanEval); a dagger marks $|\Delta|$ below $1.96\times$ the binomial standard error of the difference at that count. The test is unpaired and therefore conservative: it resolves 18 of the 28 cells (17 of the 27 leads, plus the single loss), whereas the paired test of Appendix J on stored per-item outcomes resolves 20 and additionally clears OpenBookQA on Mixtral. *Comparability.* Avg(7) is the same axis as Table 1 of BitsMoE, which is what makes the ratio comparison of Appendix I possible. *Provenance.* The three generative tasks were measured as evaluation-only reruns on the saved checkpoints, so they exist for the three configurations shown; an independent harness reproduces those columns once two of its conventions are accounted for (Table 22).

Table 15: Kernel time and L2 gather bandwidth versus codebook size (4096×6144, idle H100; kernel and shape fixed).

Codebook	Size	Kernel time	L2 gather	vs. $K = 2^{12}$
$K = 2^{12}$	0.06 MiB	82.9 μs	607 GB/s	—
$K = 2^{14}$	0.25 MiB	108.5 μs	464 GB/s	+30.9 %
$K = 2^{16}$ (ours)	1.00 MiB	133.3 μs	378 GB/s	+60.9 %

K.2 Weight memory

K.3 Coding dimension at a fixed rate

Table 24 is the ladder of §2.1. The three rungs are rate-matched exactly, not only in index cost: because the codebook carries no offset at any dimension, each rung stores one group scale and the same rank-14 factors, so all three sit at 2.340 bit/weight on DeepSeek and at that model's rate elsewhere. The step from $v = 1$ to $v = 4$ is worth 13.1 points of excess perplexity on average and the step from $v = 4$ to $v = 8$ a further 4.1, so the return on dimension is substantial but diminishing; $v = 16$ at this index rate would require $K = 2^{32}$, so the ladder stops here (§3.2).

K.4 Scaling

The FP16-relative perplexity ratio lies between 1.084 and 1.230 from 14B to 250B and is not monotone in scale: the smallest degradation is at 250B (1.084) and the largest at 16B (1.230); the mean of the two large models, 1.134, is below the mean of the four small and mid-sized ones, 1.165. A natural interpretation is that in large MoE models with many thin experts the quantization error of one expert is averaged out by the top-k sum; since scale and expert configuration vary together across the six models, the claim is limited to degradation not growing with scale.

K.5 Component ablation and rate–distortion placement

Table 26 is the component ablation discussed in §4.4: codebook size, low-rank rank, group size and router precision, each varied alone, with the value per bit of each axis in the last block. The frontier of Figure 4b shows that even at the same effective bit rate the placement of components matters: at about 2.1 bit/weight, removing low-rank compensation and spending those bits on the codebook beats the opposite arrangement by 9.3% on DeepSeek and 3.8% on Qwen3-30B. Mixtral has no pairable setting in this range and is omitted.

K.6 Where should the bits go?

Table 26 varies one component at a time and reports the quality gained per bit. Raising the codebook size from 2^{12} to 2^{16} improves perplexity and routing agreement monotonically on all three models and flattens at 2^{16}, which is also a design ceiling: the index fits 16 bits exactly and the codebook stays in L2. Raising the low-rank rank improves perplexity at a cost that differs greatly by model: the value per bit of $r{=}0 \to 14$ is 9.7 (DeepSeek), 13.2 (Qwen3-30B) and 78.3 points per bit/weight (Mixtral), an 8.0$\times$ variation, and on Mixtral it exceeds even enlarging the codebook (60.2). The cause is the cost term $16r(O + I)/(O\,I)$: on Mixtral's large inner dimension the same rank spends far fewer bits. Component priority is therefore shape-dependent, low-rank compensation first for thick experts and the codebook first for thin and numerous ones. On Qwen3-30B the successive differences among ranks 8, 14 and 20 (0.37–0.55%) fall below its 0.83% spread and are not ranked. Group size has its knee at $g = 128$ (22.1 points per bit from 256 to 128, 6.4 from 128 to 64; DeepSeek only). For the router, whose full precision is common practice at a price that had not been quantified, a four-point sweep shows that a 2-bit router costs 13.9% in perplexity and drops agreement from 0.807 to 0.634 to save 0.02% of the parameters; that the two metrics move together also supports the routing metric of §4.4. Figure 4 shows that at an equal rate of about 2.1 bit/weight, spending bits on the codebook rather than on low-rank compensation is 9.3% (DeepSeek) and 3.8% (Qwen3-30B) better.

Table 16: K-EXAONE-236B-A23B under FP16, W4A16 and 2-bit MoE-VQ: IFEval (prompt-level strict, 541 prompts), KMMLU (45 subjects × 100), HAE-RAE (2,941) and GSM8K (8-shot CoT, 250 problems).

Metric	FP16	W4A16	MoE-VQ 2-bit (ours)	2-bit retention
IFEval (prompt-strict)	0.858	0.834	0.826	96.3%
KMMLU	0.600	0.597	0.546	91.0%
HAE-RAE	0.402	0.395	0.394	98.0%
GSM8K (8-shot CoT)	0.864	0.840	0.820	94.9%

K.7 The codebook does not trade against low-rank compensation

Table 27 measures the direction that §5 closes. Shrinking the codebook from 2^{16} to 2^{12} takes the lookup table from 1 MiB to 64 KiB and cuts kernel time by 37.8% (Table 15), but spending the freed bits on rank to hold the effective rate fixed makes perplexity worse on every model. The matching rank is set by shape: because the low-rank term is $16r(O + I)/OI$, Mixtral's wide experts need $r = 122$ to match the rate while Qwen3-30B's narrow ones match at $r = 37$. The largest losses are nonetheless Mixtral's ($+34.0\%$) and DeepSeek's ($+35.4\%$). The value-per-bit figures of Table 26 are local slopes measured over $r{=}0 \to 14$ and $K{=}2^{14} \to 2^{16}$; this table shows how quickly rank saturates outside that range.

K.8 Relation to rotation

QuIP# and the QuaRot family decorrelate coordinates with a Hadamard rotation before quantizing. In a same-batch paired comparison the improvement on Mixtral is 4.7× the spread and on Qwen3-30B 1.8×, both significant; DeepSeek is within spread and only Qwen1.5 worsens significantly (3.5×). RHT is thus a gain on two models, neutral on one and a loss on one. The reason for excluding it from the default is a trade-off: a Hadamard transform must be applied at every inference while the gain is model-dependent, between 0 and 1.6%. The Gaussian codebook and the rotation appear to overlap partially, since rotation drives the coordinate distribution toward an isotropic Gaussian and §2.2 shows that group-normalized expert weights are already close to that state.

K.9 Robustness of the comparison

A different perplexity corpus. The conclusion does not change on WikiText2. The best scalar result is a mean ratio of 1.267 (RHT+GPTQ) and the VQ family goes below it, MoE-VQ at 1.218 and with RHT at 1.194. The Spearman correlation of the method ranking between corpora is 0.84–0.98 across models, and the method attaining the minimum changes on only one model, Qwen1.5 (MoE-VQ on C4, RHT+MoE-VQ on WikiText2, both variants of our method).

Number of calibration samples. Quadrupling the samples improves C4 perplexity by 1.33%; our improvement over tuned GPTQ on the same model is 8.67%, so the sample count is a second-order effect about one sixth of the method gap. The improvement also stays within the calibration domain: only C4 perplexity improves monotonically, and both changes on WikiText2 are within the spread of Table 32.

Calibration domain. AlphaQ's objection to calibration-based methods is that a calibration set is an imperfect proxy and expert activation frequencies differ by domain. Holding everything else fixed and calibrating on WikiText2 instead of C4 worsens C4 perplexity by 1.4–4.5%, against method gaps of 9–22%, so the cost of domain mismatch stays well below the method gap on every model. The swing is smaller for MoE-VQ than for GPTQ on all four models (by 1.45, 2.78, 1.97 and 2.74×; 1.87% versus 4.00% on average): the method depends less on the calibration domain, consistent with a codebook that sees neither the model nor the corpus, leaving the block-LDLQ Hessian as the only channel. The WikiText2 swing is unusually large on Mixtral (18.71% for GPTQ, 10.39% for MoE-VQ): with eight experts the same 128 sequences give far more tokens per expert, and its FP16 WikiText2 perplexity of 3.84 magnifies relative change; the direction and the ordering of the two methods are unchanged. This effect is of the same order as the sample-count effect (1.33%); both axes of calibration design are an order of magnitude below the method gap.

Table 17: C4 and WikiText2 perplexity of K-EXAONE-236B-A23B at the same quantization configuration: the reported H100 run, a second H100 run, and a rerun on B200 (sm_100) with a different CUDA/PyTorch build and device map.

Run	Hardware	C4	WikiText2
reported (Table 25)	H100 (sm_90)	24.1323	14.0750
replicate, same configuration	H100 (sm_90)	23.9629	13.9833
rerun	B200 (sm_100)	24.2802	14.1440
spread between the two H100 runs		+0.71%	+0.66%
difference across hardware		+0.61%	+0.49%

Replicate spread. WikiText2 spread exceeds C4 spread, by 8× on DeepSeek, so differences between rows of Table 29 must be read against a looser standard than those of Table 1; this is why C4 is the primary metric, and it applies equally when comparing with prior work that reports WikiText2.

Table 18: WikiText2 perplexity at two bits relative to each paper's own FP16 (lower is better); KBVQ-MoE numbers from their Table 1.

	KBVQ-MoE paper (2.08 bit/weight)			This paper		
Model	GPTQ	VQ dim-4	KBVQ-MoE	bpw	tuned GPTQ	MoE-VQ (ours)
Qwen1.5-MoE	1.758	3.733	1.331	2.34	1.232	**1.099**
DeepSeek-V2-Lite	1.434	1.851	1.341	2.34	1.286	**1.174**
Qwen3-30B-A3B	50.393	13.253	1.364	2.43	1.339	**1.179**
Mixtral-8x7B	1.466	1.544	**1.389**	2.19	1.879	1.420

Table 19: Accuracy retention relative to each paper's FP16 on the two overlapping tasks.

	HellaSwag		WinoGrande	
Model	KBVQ-MoE	MoE-VQ (ours)	KBVQ-MoE	MoE-VQ (ours)
Qwen1.5-MoE	0.835	**0.966**	0.940	**1.008**
DeepSeek-V2-Lite	0.872	**0.915**	0.954	**0.991**
Qwen3-30B-A3B	0.848	**0.957**	0.943	**1.010**
Mixtral-8x7B	0.938	**0.951**	0.963	**0.991**

Table 20: Our harness against the official lm-evaluation-harness 0.4.13 on FP16 (zero-shot, `transformers` 5.14.1 for both). SE is the official standard error; the last column gives $|\Delta|$ in units of SE.

Model	Task	Official	Ours	Δ	SE	$\lvert\Delta\rvert$/SE
DeepSeek-V2-Lite	HellaSwag	73.43	74.78	$+1.35$	0.44	3.1
	MMLU	49.81	49.66	-0.15	0.41	0.4
	OpenBookQA	43.00	44.00	$+1.00$	2.22	0.5
	WinoGrande	67.32	67.17	-0.15	1.32	0.1
Qwen3-30B-A3B-Base	HellaSwag	81.42	81.39	-0.03	0.39	0.1
	MMLU	78.68	78.67	-0.01	0.32	0.0
	OpenBookQA	44.80	44.40	-0.40	2.23	0.2
	WinoGrande	73.01	72.77	-0.24	1.25	0.2

Table 21: Saved 2-bit checkpoints re-scored with the official lm-evaluation-harness 0.4.13. *paired* Δ: MoE-VQ minus tuned GPTQ, exact McNemar test on the same items ($^{*}p < 0.05$, $^{**}p < 10^{-3}$); ‡: void (Mixtral tokenizer artifact, Table 22). GSM8K uses the flexible-extract filter.

Model	Method	HellaS.	MMLU	OpenB.	WinoG.	MathQA	GSM8K	HumanE.
Qwen1.5-MoE	GPTQ (tuned)	71.25	52.61	40.20	68.19	30.92	33.51	14.02
	MoE-VQ (ours)	73.31	57.75	42.20	68.67	35.54	50.11	26.22
	paired Δ	$+2.06^{**}$	$+5.14^{**}$	$+2.00$	$+0.47$	$+4.02^{**}$	$+16.60^{**}$	$+12.20^{**}$
DeepSeek-V2-Lite	GPTQ (tuned)	64.72	36.57	40.60	66.22	28.27	5.61	6.10
	MoE-VQ (ours)	67.75	42.08	40.60	66.93	30.15	9.55	10.37
	paired Δ	$+3.03^{**}$	$+5.51^{**}$	$+0.00$	$+0.71$	$+1.14$	$+3.94^{**}$	$+4.27^{*}$
Qwen3-30B-A3B	GPTQ (tuned)	75.80	69.44	41.40	71.82	51.42	73.46	24.39
	MoE-VQ (ours)	77.83	75.25	43.80	73.56	49.21	80.59	39.02
	paired Δ	$+2.03^{**}$	$+5.81^{**}$	$+2.40$	$+1.74$	-2.38^{*}	$+7.13^{**}$	$+14.63^{**}$
Mixtral-8x7B	GPTQ (tuned)	60.51	47.74	35.00	64.96	26.63	11.98	0.00
	MoE-VQ (ours)	79.86	62.21	45.80	77.90	37.79	34.87	0.61
	paired Δ	$+19.36^{**}$	$+14.47^{**}$	$+10.80^{**}$	$+12.94^{**}$	$+10.99^{**}$	$+22.90^{**}$	$+0.61^{\ddagger}$

Table 22: HumanEval pass@1 (%) of the official harness's generations under successively more permissive scoring: *+indent* restores the leading space dropped by Mixtral's tokenizer; *+fence* also treats a markdown fence as a stop sequence. *ours*: our harness (Table 2). Last column: completions that begin with three spaces.

Model	Method	official	+indent	+fence	ours	3-space
Qwen1.5-MoE	GPTQ (tuned)	14.02	14.02	14.02	14.63	0/164
	MoE-VQ (ours)	26.22	26.22	26.22	27.44	0/164
DeepSeek-V2-Lite	GPTQ (tuned)	6.10	6.10	6.10	6.71	0/164
	MoE-VQ (ours)	10.37	10.37	10.37	12.20	0/164
Qwen3-30B-A3B	GPTQ (tuned)	24.39	24.39	28.66	26.83	0/164
	MoE-VQ (ours)	39.02	39.02	52.44	54.27	0/164
Mixtral-8x7B	GPTQ (tuned)	0.00	0.00	0.00	0.00	104/164
	MoE-VQ (ours)	0.61	24.39	24.39	25.00	163/164

Table 23: Weight memory (decimal GB). (A): MoE-VQ experts at 2 bits, shared experts at 4 bits, all else FP16, the configuration of every quality result; (B): (A) with W4 attention and dense MLPs. GPU counts are capacity minima against the 85.5 GB usable on an 80 GB H100.

Model	Params	FP16	W4A16	(A) verified	(B) deploy	GPUs (FP16 → A / B)
Qwen1.5-MoE-A2.7B	14.3B	28.6	8.5	6.14	5.55	1 → 1 / 1
DeepSeek-V2-Lite	15.7B	31.4	9.0	6.17	5.53	1 → 1 / 1
Qwen3-30B-A3B-Base	30.5B	61.1	17.1	11.88	10.55	1 → 1 / 1
Mixtral-8x7B-v0.1	46.7B	93.4	25.2	15.59	13.62	2 → 1 / 1
K-EXAONE-236B-A23B	236.6B	473.1	128.5	79.77	71.29	6 → 1 / 1
Solar-Open2-250B	250.3B	500.6	135.3	85.63	77.20	6 → 2 / 1

Table 24: C4 perplexity (excess over FP16 in parentheses) versus coding dimension v at the index rate $\log_2 K/v = 2.000$, with one group scale, block-LDLQ feedback and the rank-14 compensator at every rung. $v = 1$ is the Lloyd–Max quantizer for $\mathcal{N}(0, 1)$.

Coding dimension	bit/weight	Qwen1.5-MoE	DeepSeek-V2-Lite	Qwen3-30B-A3B	Mixtral-8x7B	Mean excess
$v = 1$, $K = 4$ (Lloyd–Max)	2.34	10.17 (+20.0%)	13.03 (+36.8%)	11.74 (+20.9%)	10.82 (+58.5%)	34.0%
$v = 4$, $K = 2^8$	2.34	9.72 (+14.6%)	12.12 (+27.1%)	11.14 (+14.8%)	8.68 (+27.0%)	20.9%
$v = 8$, $K = 2^{16}$ (MoE-VQ, ours)	2.34	**9.55 (+12.6%)**	**11.74 (+23.2%)**	**10.88 (+12.0%)**	**8.15 (+19.3%)**	**16.8%**
Reduction, v=1 → v=8		7.4 pp	13.5 pp	8.9 pp	39.1 pp	17.2 pp
in units of replicate spread		74×	113×	11×	115×	—

Table 25: C4 perplexity ratio and Avg_4 retention relative to FP16 at two bits, by model (236B and 250B in Appendix H).

Model	Params	Experts	PPL_{FP16}	$\text{PPL}_{\text{2-bit}}$	Ratio↓	Acc_{FP16}	Retention
Qwen1.5-MoE-A2.7B	14B	60	8.48	9.54	1.124	62.90	96.3 %
DeepSeek-V2-Lite	16B	64	9.53	11.73	1.230	58.90	93.1 %
Qwen3-30B-A3B-Base	30B	128	9.71	10.84	1.117	69.31	97.0 %
Mixtral-8x7B-v0.1	47B	8	6.83	8.13	1.190	68.32	95.5 %
K-EXAONE-236B-A23B	236B	128	20.37	24.13	1.185	60.41	94.1 %
Solar-Open2-250B	250B	320	9.35	10.13	**1.084**	72.47	**98.0 %**
Mean of four small/mid models					1.165		95.5 %
Mean of two large models					**1.134**		96.1 %

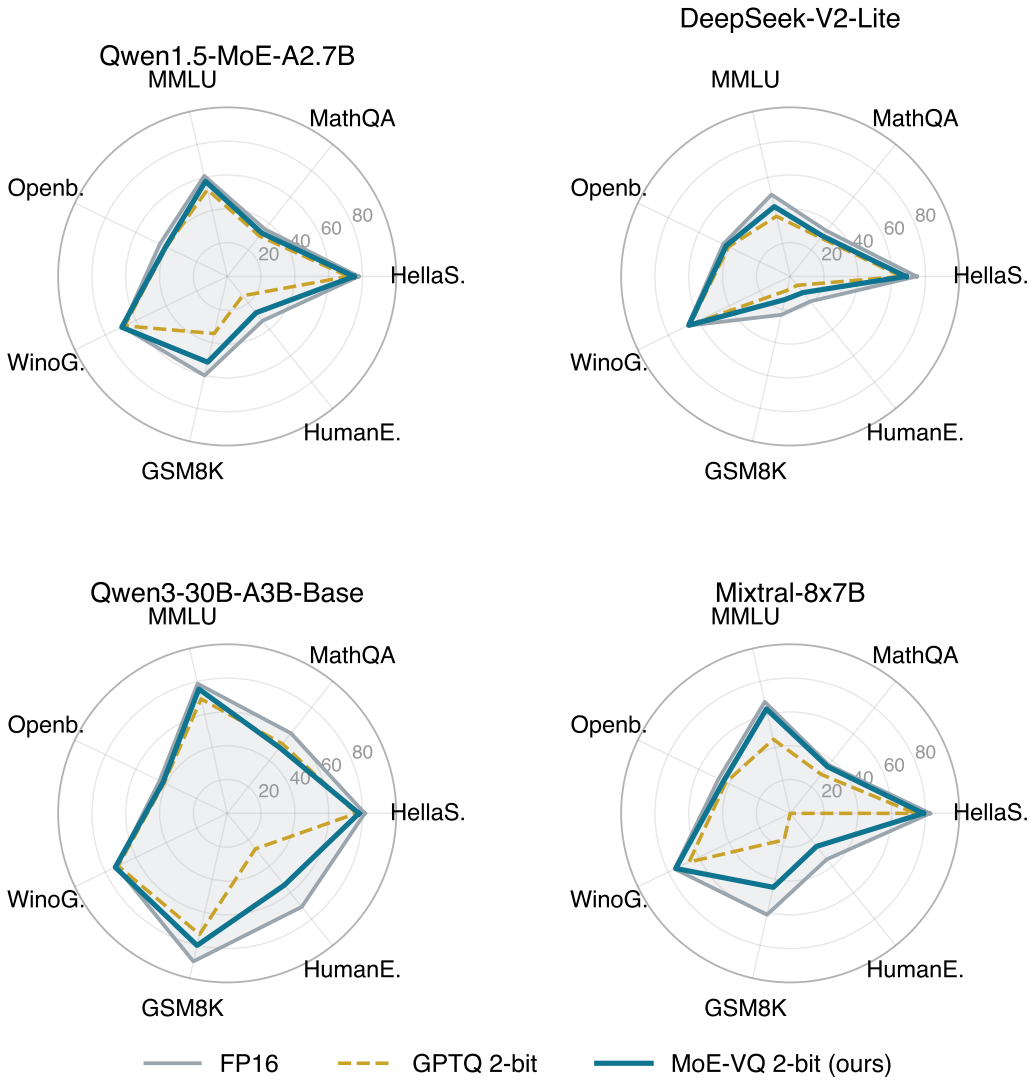


Figure 3: Table 2 as a shape: seven-task accuracy of FP16, tuned GPTQ and MoE-VQ at two bits, one panel per model. MoE-VQ holds the FP16 contour where tuned GPTQ collapses toward the centre. The largest single gap is a generative axis on three of the four models (GSM8K $+17.4$ on Qwen1.5, HumanEval $+27.4$ on Qwen3-30B, GSM8K $+28.8$ on Mixtral) and MMLU $+5.9$ on DeepSeek.

Table 26: Component ablation at two bits: C4 perplexity ratio to FP16 and routing Jaccard, each component varied alone from $K = 2^{16}$, $g = 128$, $r = 14$ (one batch, shared experts at the routed rate). Value per bit: improvement in ratio (points) per bit/weight.

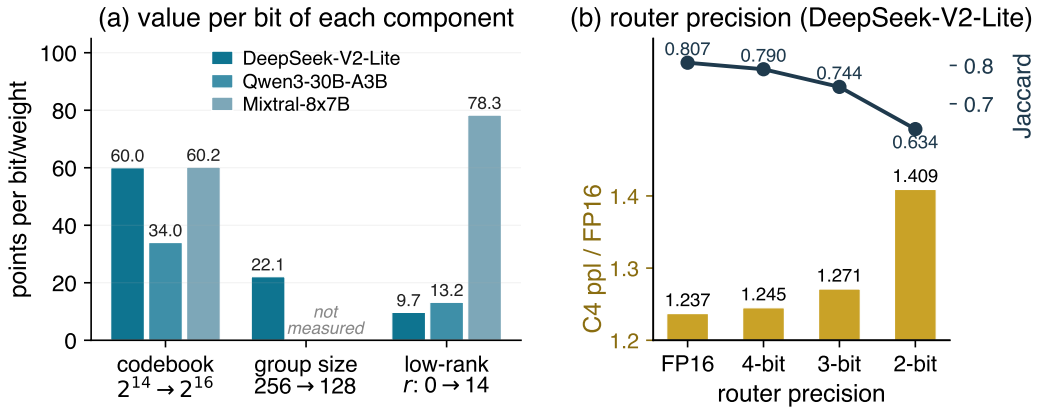
		DeepSeek-V2-Lite			Qwen3-30B-A3B			Mixtral-8x7B		
Axis	Setting	bpw	Ratio↓	Jacc.	bpw	Ratio↓	Jacc.	bpw	Ratio↓	Jacc.
(a) Codebook K	2^{12}	1.840	1.690	0.723	1.929	1.395	0.766	1.690	1.725	0.805
	2^{14}	2.090	1.387	0.768	2.179	1.203	0.807	1.940	1.341	0.847
	2^{16}	2.340	**1.237**	**0.807**	2.429	**1.118**	**0.837**	2.190	**1.190**	**0.873**
(b) Low-rank r	0	2.125	1.258	0.803	2.125	1.158	0.816	2.125	1.241	0.854
	4	2.187	1.245	0.802	—	—	—	—	—	—
	8	2.248	1.245	0.803	2.299	1.124	0.835	2.162	1.191	0.873
	14	2.340	1.237	0.807	2.429	1.118	0.837	2.190	1.190	0.873
	20	2.433	**1.232**	**0.809**	2.559	**1.114**	0.837	2.218	**1.188**	0.871
(c) Group g	256	2.278	1.251	0.806	not measured					
	128	2.340	1.237	0.807						
	64	2.465	**1.229**	**0.811**						
(d) Router	FP16	2.340	**1.237**	**0.807**	not measured					
	4-bit	2.340	1.245	0.790						
	3-bit	2.340	1.271	0.744						
	2-bit	2.340	1.409	0.634						
Value per bit (points/bpw)										
codebook $2^{14} \to 2^{16}$			60.0			34.0			60.2	
group $256 \to 128$			22.1			n/m			n/m	
low-rank $r0 \to r14$			9.7			13.2			**78.3**	

Table 27: Two configurations at the same effective rate: $K = 2^{16}$ with $r = 14$, and $K = 2^{12}$ with the rank that matches the rate. ΔC4: relative change in C4 perplexity.

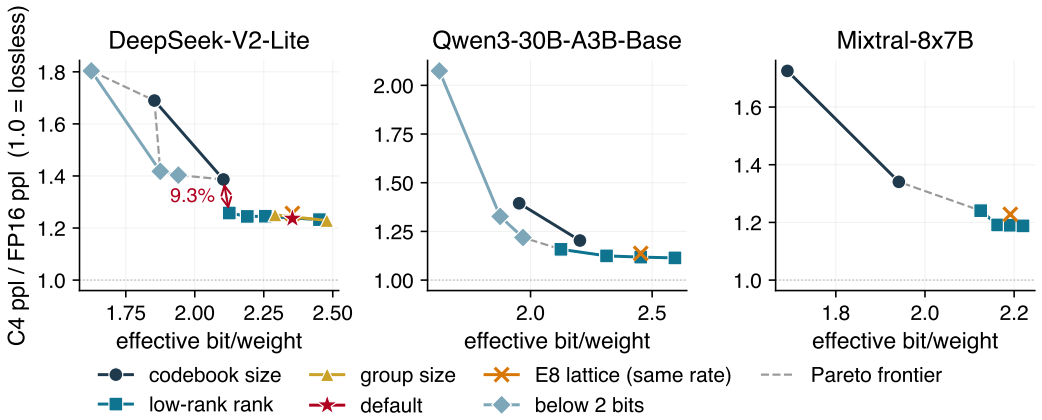
Model	Configuration	bit/weight	C4	WikiText2	ΔC4
Qwen1.5-MoE	2^{16}, r=14	2.340	9.5662	7.9222	
	2^{12}, r=46	2.333	11.1814	9.0795	+16.9%
DeepSeek-V2-Lite	2^{16}, r=14	2.340	11.7298	8.5935	
	2^{12}, r=46	2.333	15.8795	11.3512	+35.4%
Qwen3-30B-A3B	2^{16}, r=14	2.429	10.8338	7.1363	
	2^{12}, r=37	2.428	12.9995	8.8354	+20.0%
Mixtral-8x7B	2^{16}, r=14	2.190	8.1439	5.4477	
	2^{12}, r=122	2.192	10.9113	8.8846	+34.0%

Table 28: C4 perplexity with and without a Hadamard rotation (RHT) in the same batch; verdicts in multiples of the replicate spread.

Model	MoE-VQ (ours)	RHT + MoE-VQ (ours)	Change	Spread	Verdict
DeepSeek-V2-Lite	11.736	11.743	+0.06 %	0.12 %	within spread
Qwen1.5-MoE-A2.7B	9.530	9.564	+0.35 %	0.10 %	worse (3.5×)
Qwen3-30B-A3B-Base	10.935	10.771	−1.50 %	0.83 %	**better** (1.8×)
Mixtral-8x7B-v0.1	8.124	7.994	−1.60 %	0.34 %	**better** (4.7×)



(a) Value per bit of each component, and the router-precision sweep.



(b) Rate–distortion frontier.

Figure 4: (a) The improvement per bit/weight of enlarging the codebook is stable across models, whereas that of low-rank compensation varies by 8.0× with matrix shape; quantizing the router (DeepSeek) costs perplexity and routing agreement while saving almost no bits. (b) At an equal effective bit rate the placement of components changes the result (arrow: two configurations at the same rate on DeepSeek).

Table 29: WikiText2 perplexity (context 2,048) of the checkpoints of Table 1. Last column: mean ratio to FP16 over the four models. Bold: best per column.

Method	Qwen1.5-MoE	DeepSeek-V2-Lite	Qwen3-30B-A3B	Mixtral-8x7B	Mean ratio↓
FP16	7.220	7.283	6.087	3.842	1.000
RTN	11.877	—	—	58.876	8.484
GPTQ (tuned)	8.895	9.370	8.152	7.219	1.434
AWQ	8.749	9.651	8.640	5.731	1.362
GPTQ + low-rank	8.210	9.160	7.663	6.548	1.339
AWQ + DRQ + low-rank	8.283	9.104	7.533	6.784	1.350
RHT + GPTQ	8.155	9.104	7.555	5.555	1.267
VQ dim-4 + low-rank	8.001	8.860	7.344	6.315	1.294
MoE-VQ w/o feedback (ours)	8.497	8.656	7.974	5.691	1.289
RHT + MoE-VQ (ours)	**7.851**	8.570	**7.107**	**5.158**	**1.194**
MoE-VQ (ours)	7.934	**8.554**	7.168	5.455	1.218

Table 30: C4 and WikiText2 perplexity versus the number of calibration sequences (DeepSeek-V2-Lite, MoE-VQ); changes are relative to 128.

n_{seq}	C4 PPL	(vs 128)	WikiText2 PPL	(vs 128)
128 (paper setting)	11.730	—	8.554	—
256	11.634	$-0.81\,\%$	8.591	$+0.43\,\%$
512	11.574	$-1.33\,\%$	8.528	$-0.31\,\%$
Reference: tuned GPTQ $\rightarrow$ MoE-VQ improvement on the same model $= 8.67\,\%$ ($6.5\times$ this effect)				

Table 31: Perplexity with C4 or WikiText2 as the calibration corpus, all else as in Table 1. *Swing*: relative change on each evaluation corpus when the calibration corpus changes.

		Calibrated on C4		Calibrated on WikiText2		Swing	
Model	Method	C4	Wiki	C4	Wiki	C4	Wiki
Qwen1.5-MoE 14B	GPTQ	11.19	8.89	11.56	8.48	$+3.25\,\%$	$+4.61\,\%$
	MoE-VQ (ours)	9.54	7.93	9.75	7.72	$\mathbf{+2.24}\,\%$	$+2.69\,\%$
DeepSeek-V2-Lite 16B	GPTQ	12.84	9.37	13.33	8.77	$+3.78\,\%$	$+6.44\,\%$
	MoE-VQ (ours)	11.73	8.55	11.89	8.20	$\mathbf{+1.36}\,\%$	$+4.09\,\%$
Qwen3-30B-A3B 30B	GPTQ	12.06	8.15	12.59	7.78	$+4.41\,\%$	$+4.59\,\%$
	MoE-VQ (ours)	10.84	7.17	11.09	6.98	$\mathbf{+2.24}\,\%$	$+2.68\,\%$
Mixtral-8x7B 47B	GPTQ	10.37	7.22	10.84	5.87	$+4.54\,\%$	$+18.71\,\%$
	MoE-VQ (ours)	8.13	5.45	8.26	4.89	$\mathbf{+1.66}\,\%$	$+10.39\,\%$
Mean C4 swing						GPTQ 4.00 %	MoE-VQ (ours) **1.87** %

Table 32: Replicate spread: the max–min range of MoE-VQ perplexity over four independent batches with identical settings.

Model	C4	WikiText2	Wiki / C4
Qwen1.5-MoE-A2.7B	$0.10\,\%$	$0.35\,\%$	$3.5\times$
DeepSeek-V2-Lite	$0.12\,\%$	$0.99\,\%$	$8.3\times$
Qwen3-30B-A3B-Base	$0.83\,\%$	$0.83\,\%$	$1.0\times$
Mixtral-8x7B-v0.1	$0.34\,\%$	$0.68\,\%$	$2.0\times$