What Is Left in Trained LLM Weights?

Exact coding of BF16 weights below their order-0 entropy, and what the remaining structure shows

David Herrera
TeamIDE Lab
david@teamide.dev

November 2026

Abstract

We asked how far exact (bit-identical) coding of a trained language model's BF16 weights can go, and what the answer says about the weights. Our codec (format v4, open source) predicts each unit of a tensor from already-decoded units of earlier tensors, or earlier rows of the same tensor, of the same model; no other model is needed. It stores the whole of Qwen3.8-27B in 10.4305 bits/weight (36.22 GB from 55.56 GB), the whole of Qwen3.6-27B in 10.4328 and the whole of Qwen2.5-32B-Instruct, a model outside their lineage, in 10.3820 (42.52 GB from 65.53 GB), and every shard rebuilds with a SHA-256 equal to the published file. On Qwen3.8-27B that is 0.062 below the per-tensor order-0 entropy (10.4924, the bound for coding each tensor's weights i.i.d. with one table), 0.18 below the DFloat11 scheme computed on the same weights, and 0.19 below ZipNN and 2.03 below zstd −3 run on the same files. The gain is small, and decoding runs on the CPU only. Charging every coefficient and table, and testing each structural claim against a shuffled null, we find little structure inside a tensor beyond row and column scale and row-to-row structure in attention projections. Most of what remains sits between tensors. A neuron's output weights are partly predictable from its own input weights, an alignment known from interpretability work. As a coding gain it is worth an estimated 0.029–0.042 bits/weight (sampled layers) in seven models from six families and peaks at 25–52% depth; in the six SwiGLU models the sign of the relation turns negative late in depth. Neurons also keep their index across releases, and the weights are consistent with Qwen3.5-27B having been grown from Qwen3.5-9B by copying channels, neurons and a 16-layer stretch; the codec uses these copies. We report the levers that did not pay, and release the data behind every figure and table.

1 Introduction

Exact compression of model weights is a measuring stick. A coder that rebuilds every bit can only save what it can predict, so every bit saved, net of the side information it needs, is structure that training left in the weights. We used that stick on real checkpoints, with three rules: every number is a net code length with all side information charged; every structural claim has a null that destroys the structure; whole-model claims come only from the codec, decoded back and checked by hash.

The question was how far below the order-0 entropy a normally trained BF16 model can be coded exactly, and what the structure that allows it looks like. We were not trying to beat a particular method. There are three results.

- **Exact coding below the per-tensor order-0 entropy, end to end.** Format v4 predicts each unit (a neuron's or channel's vector) from up to 8 already-decoded units of earlier

tensors or earlier rows of the same model, in integer arithmetic. Whole Qwen3.8-27B: 10.4305 bits/weight, 0.062 below its order-0 entropy of 10.4924; whole Qwen3.6-27B: 10.4328; whole Qwen2.5-32B-Instruct, outside that lineage: 10.3820. All shards rebuild bit-identical. On the same Qwen3.8 files ZipNN measures 10.6155 and zstd −3 12.4590 (§5). The gain over simpler exact coding is small (§8).

- **Where the remaining structure is.** Inside a tensor there is little: row and column scale (0.014 bits/weight) and row-to-row structure in attention projections (§4, App. A). Between tensors, a neuron's output vector is partly predictable from its own input vectors. This alignment has been studied in interpretability work [11, 15–17]; we measure what it is worth in bits: 0.029–0.042 bits/weight model-wide in 7 models from 6 families (Gaussian estimate, sampled layers), peaking at 25–52% depth, with the sign of the down–up relation turning negative in the last 6–33% of depth in every SwiGLU model (§6).
- **Lineage and growth, read from the weights.** Neurons of Qwen3.5/3.6/3.8-27B align index for index. The weights are consistent with Qwen3.5-27B having been grown from Qwen3.5-9B by copying channels, neurons and a 16-layer stretch. On OLMo-2-7B's public checkpoints, a positive control, the same test recovers the training order (§7). The copies are coding sources that v4 uses.

Further results are in the appendices: the levers that did not pay (App. A), coding with a related model as side information (App. D), exact coding of Q8_0 and K-quant files, and a small lossy quantization result (App. E).

2 Background and related work

BF16. A BF16 weight has 1 sign, 8 exponent and 7 mantissa bits [24]. Trained weights cluster near zero, so the exponent uses few values and has low entropy. The mantissa is close to uniform.

Lossless weight compression. Entropy coding of network weights goes back to Huffman coding of quantized weights [18] and context-adaptive arithmetic coding [43]; these code quantized values, not exact floats. For exact BF16, DFloat11 [53] Huffman-codes the exponent and stores sign and mantissa raw. Its table reports 67.58–68.17% of the original size, or 10.81–10.91 bits/weight, on nine LLMs, and it decodes on the GPU. ZipNN [21, 22] groups bytes and Huffman-codes exponents (BF16 at 66.3–67.4%, about 10.6–10.8 bits/weight). NeuZip [19] uses ANS on exponents. Huff-LLM [51] measures an order-0 entropy of 10.54 bits/weight for Llama-3-8B FP16 weights and Huffman-codes a 1-5-5-5 bit split (entropy 10.61) in hardware. Heilper and Singer [20] code exponent and mantissa streams per tensor (OPT-1.3B BF16 at 66.7%, 10.67 bits/weight). ECF8 [50] treats FP8; ZipServ, ENEC and TRACE [12, 45, 49] target fast decode on accelerators or memory links. A 2026 study [36] measures order-0 entropy per tensor, finds ANS within 0.1–0.2 bits of it on BF16 (its §VI), and calls it the bits per weight "required under any lossless encoding scheme" (its §II-B). All of these code each tensor's weights independently, with one distribution per tensor or field, so the per-tensor order-0 entropy bounds them. The lowest coded BF16 figure we found was ZipNN's 66.3% (≈10.6 bits/weight); the lowest reported order-0 entropy, Huff-LLM's 10.54. We found no coder that predicts across tensors of one model. Our search was not exhaustive.

Coding against another model. ZipLLM/BitX [42] deduplicates checkpoints and XOR-codes fine-tunes against their base. FM-Delta [33] entropy-codes the integer difference between a fine-tune

and its base; TensorDex [28] picks per-tensor bases from other models; ZipNN and Heilper and Singer also code checkpoint deltas, and Kim and Belyaev [26] use the previous checkpoint as context. App. D measures this setting on our models.

Prediction across layers. MCWC [27] aligns hidden units and heads across depth (motion compensation), then predicts each layer from the previous one with keyframes and a learned entropy model, lossily. An earlier lossy scheme predicts weights from the previous layer under a smoothness hypothesis [30]. v4's search for partners in layer $L-1$ is a lossless relative of MCWC's alignment step. D^{2}Quant [46] folds a per-column scale of the down projection into the up projection's rows. GPTQ, SpQR, QuIP# and AQLM [7, 10, 13, 39] quantize one matrix at a time.

Input–output alignment of neurons. Elhage et al. [11] suggested that neurons with opposite input and output directions clear features from the residual stream. Gurnee et al. [17] measure $\cos(w_{\text{in}}, w_{\text{out}})$ per neuron by layer in GPT-2 and Pythia-160M; high-frequency neurons, mostly in the first quarter of depth, have nearly opposite input and output weights. Gerstner and Schütze [15, 16] classify gated neurons in GLU models by the cosines between their input, gate and output weights: "strengthening" neurons dominate early and middle layers and "weakening" ones the late layers. Voita et al. [40] find token-detecting neurons whose updates remove information about the tokens that trigger them. Section 6 measures the same alignment as a coding gain.

Lineage and growth. Lineage can be tested from weights alone: HuRef [52] identifies a base model from parameter directions; Zhu et al. [54] test independence with permutation nulls and locate the Llama-3.1-8B layers that each Llama-3.2 layer derives from; GhostSpec [41] detects reuse from spectral signatures; Witness Overlap [31] infers the direction of derivation inside open-weight families. Net2Net [4] widens and deepens networks by copying units; SOLAR [25], LLaMA Pro [44] and stacking [8] grow LLMs in depth. Section 7 applies tests of this kind to the Qwen 27B models and maps a width-and-depth growth step.

3 Method: how the numbers were made

Net code length. Every saving is reported in bits per weight of the tensor or model named. It is net of every coefficient, scale, table, index and header. Research measurements get a code length in one of three ways: an adaptive coder, which includes the cost of learning its model; static context tables, which include the table cost; or the ideal code length under a fitted Gaussian, integrating its mass over each BF16 rounding interval, with its parameters charged. Some per-layer estimates use the Gaussian approximation $-\frac{1}{2}\log_2(1-\rho^2)$ per element. Where we compared it with a real coder, the coder realized 0.90 of the Gaussian figure (Qwen3.8 down projections). We label which method each number comes from.

Nulls. Every structural claim has a null that keeps the statistics but destroys the claimed relation: pairing a unit with another unit's vectors, permuting channels, shifting indices, permuting base rows or shuffling tokens. A claim stands only if the null sits at its floor, the charged side information. Where a direction shared by all units raises the null above that floor, we subtract the excess (named where it happens); this assumes the shared part adds linearly.

Fitting. Per-unit coefficients are fit in sample and charged, which is a valid two-part code. Refitting on even elements and scoring on odd ones lowers the net by at most 0.008 per layer. The per-neuron residual width ($\sim$8 bits per neuron, $\sim$0.0003 bits/weight model-wide) is not charged. Learned predictors are fit on some layers and scored on others.

Red tests. Every tool must find planted structure and report zero on noise. The entropy tools match exact truths: an i.i.d. Gaussian measures 10.4636 against the exact 10.4638, and a Laplace 10.7756 against 10.7761. Those tests pass 8/8. In 13/13 planted-structure tests the structure is found, and noise reads zero. Each later study adds its own suite (for example growth and lineage 28/28, levers 22/22 and 24/24). We also mutate the source and check that the suite fails before we trust a pass.

End to end. A whole-model number counts only if it comes from the codec. Each shard is encoded and decoded back from the written file. The rebuilt shard's SHA-256 must equal the Hugging Face LFS hash. Every byte of every output file is counted. Sampled measurements extrapolated to a whole model are labelled *sampled* or *projection*, and are kept apart from end-to-end numbers.

Models. Qwen3.8-27B BF16 (1,199 tensors, 27,781,427,952 weights, 55.56 GB) is the test bed. It has no close fine-tune base among the models we had (App. D codes it given Qwen3.5-27B). Qwen3.6-27B, of the same lineage, and Qwen2.5-32B-Instruct [47], an earlier Qwen generation outside it (64 layers, full attention in every layer, 771 tensors, 32,763,876,352 weights, 65.53 GB), are the other whole models; Table 3 lists the cross-family models [2, 3, 23, 32, 37]. Apart from the three whole models, Qwen3-0.6B (lossy test) and the GGUF files, data came from HTTP range reads of single tensors at pinned revisions. App. H lists the models with their repositories and revisions; Qwen3-0.6B and the GGUF files were used without a recorded revision.

4 The order-0 entropy, and what is not inside a tensor

Table 1 splits the order-0 cost of Qwen3.8-27B by field. The exponent carries most of the redundancy (2.57 of its 8 bits). Given the exponent and the higher mantissa bits, only the top two mantissa bits skew (0.946 and 0.984 bits); the low five are at least 0.996 bits each, and the sign is exactly 1.000. The order-0 entropy of the whole 16-bit symbol, per tensor and weighted by weights, is 10.4924. The DFloat11 scheme (Huffman exponent, 8 raw bits) measures 10.6096 on the same weights. Of its 0.12 extra, 0.08 comes from storing sign and mantissa raw (0.05 of it the exponent–mantissa dependence, 0.03 the mantissa's own skew) and 0.04 from Huffman's integer code lengths.

We expected trained weights to be heavy-tailed, near the Laplace value of 10.78, as several compression papers assume [36, 50]. Then a scale context could take them from about 10.8 to 10.4. On Qwen3.8 the weights are instead close to Gaussian per tensor, as the normal-weights view behind NF4 assumes [6]: median excess kurtosis is 0.66, and the per-tensor order-0 median is 10.494, only 0.03 above the Gaussian reference. The Gaussian value is a reference, not a floor: other shapes with the same variance can cost less. Nor is the picture universal: Qwen3.6-35B-A3B's attention and router tensors sit at 10.60–10.68 (heavier row and column scale spread). A row-by-column scale context on Qwen3.8, with its side information charged, gives 10.4785 (-0.014); it codes weights independently but not identically, so it can go below the per-tensor order-0 entropy. Neighbour and sign contexts on top of that add cost; the richest context model we built (scale, signed neighbours, sign bias) costs 0.001–0.002 more than scale alone. In the MLP tensors we tested, after scale, the weights behave like independent draws from one near-Gaussian shape. The exception is row-to-row

Field	Stored bits	Information (bits)
Sign	1	1.000
Exponent	8	2.568
Mantissa	7	6.972 (6.924 given the exponent)
Whole 16-bit symbol, order-0 per tensor	16	**10.4924**
DFloat11 scheme (Huffman exponent + 8 raw)		10.6096
Exact i.i.d. Gaussian, known scale		10.4638
Exact i.i.d. Laplace, known scale		10.7761

Table 1: Qwen3.8-27B BF16, whole model, per-tensor values weighted by weights (no table cost charged). Gaussian and Laplace references enumerate every BF16 rounding bin exactly (Gaussian at $\sigma = 0.02$).

structure in attention projections (RoPE pairs and the output-gate rows' shared direction), which v4 uses. Many other levers were coded and gave nothing net (App. A): calibration data, low-bit forensics, low rank, tensor-network folds, image predictors ($\leq$0.013 on one of four tensors), same-index layer deltas, lattice shaping, pseudo-random masks, bits-back coding and exact repeats.

5 The codec

The codec (Foundry, `ingot/`) is a container of per-tensor payloads with a directory, a source hash and per-tensor hashes. Each tensor uses whichever mode is smallest.

Plain mode (v1/v2). The 10 bits of exponent and top two mantissa bits of each weight are interleaved-rANS coded [9] with one static table per tensor (per expert for MoE); the sign and low 5 mantissa bits are stored raw. Blocks of 2^{18} weights with an offset table allow random access and parallel decode. On Qwen3.8 the whole model is 10.5076 bits/weight, against an order-0 prediction plus layout cost of 10.503.

Predict mode (v3). The same-unit rules of §6: down given gate and up, up given gate, linear-attention z given v, out given v and z, o given v. Each unit gets int16 coefficients, a per-tensor exponent and an i8 unit scale. The context is the scale class times 12 buckets of $|p|$. A binary symbol codes whether the sign of w agrees with the sign of p; then the 10-bit symbol is coded in that context and the low 5 mantissa bits are stored raw. Everything that selects a context is integer arithmetic, so encoder and decoder cannot disagree. Whole model: 10.4722 bits/weight (-0.0354).

Match mode (v4). Each unit is predicted from up to $K \leq 8$ partner units, each from any earlier tensor. A partner can be an earlier entry of the file, the tensor's own earlier rows, or a tensor of an earlier shard, referenced by name, shape and hash. The predictor is $p = \sum_k q_k X_k$ in int64, with $|q_k| < 2^{15}$ and $|X_k| \leq 2^{40}$, so $|p| < 2^{58}$. The sum cannot overflow, which makes it exact and independent of summation order: any device that sums exactly gets the same p. Only sign(p) and the bucket of $|p|$ are used, with v3's contexts and weight coder. Partner links, indices and step-coded coefficients go in one rANS block per tensor, guarded by a hash. A shard refers only to shards that sort before it, so decoding in name order always has its references.

The encoder search. Encoding is a search, not part of the format; the decoder reads every choice from the file. The encoder sketches every unit with a 512-dim ± 1 projection, keeps the top

BF16 checkpoint	Bytes	bits/weight	% of BF16
Qwen3.8-27B			
Source (BF16 safetensors)	55,563,006,776	16	100
zstd −3	43,266,226,729	12.4590	77.87
ZipNN 0.5.4	36,864,268,616	10.6155	66.35
DFloat11 scheme (computed, no index)		10.6096	66.31
Order-0 entropy per tensor (no table cost)		10.4924	
v2 plain	36,489,525,136	10.5076	65.67
v3 same-unit predict	36,366,627,032	10.4722	65.45
v4 match	**36,221,798,920**	**10.4305**	**65.19**
Qwen3.6-27B			
Source (BF16 safetensors)	55,563,006,400	16	100
Plain mode only (v2 coding, same container)	36,492,298,208	10.5084	65.68
v4 match	**36,229,587,760**	**10.4328**	**65.20**
Qwen2.5-32B-Instruct			
Source (BF16 safetensors)	65,527,841,856	16	100
DFloat11, published (its v1 table; not run by us)	45.53 GB	11.12	69.48
Plain mode only (v2 coding, same container)	43,303,666,736	10.5735	66.08
v4 match	**42,519,262,864**	**10.3820**	**64.89**
Qwen3.8-27B, shards 5, 11, 17			
zstd −3	4,918,248,690	12.4575	77.86
zstd −19	4,785,024,961	12.1200	75.75
xz −9	4,422,898,456	11.2028	70.02
ZipNN 0.5.4	4,190,501,582	10.6142	66.34
Plain mode only	4,147,282,464	10.5047	65.65
v4 match	**4,112,806,168**	**10.4174**	**65.11**

Table 2: Whole-model results, and three Qwen3.8 shards for the slow compressors. Codec rows are end to end: every byte of every output file is counted, every tensor decodes bit-identical (1,199 in each Qwen3 model, 771 in Qwen2.5), and 18/18 (Qwen3.8), 15/15 (Qwen3.6) and 17/17 (Qwen2.5) rebuilt shards have SHA-256 equal to the Hugging Face LFS hash; the plain-only sizes are the encoder's per-tensor sizes with match off, not separately written files. The DFloat11 Qwen2.5 row is copied from its paper (arXiv v1, Table 2), not measured by us. zstd and ZipNN were run by us on the same 18 files (ZipNN round trip checked); zstd −19 and xz −9, being slow, on three 2.1 GB shards (5, 11, 17: early, middle and late layers), the last block, where the two Ingot rows are the per-shard sizes from the same v4 run (a v4 shard is coded given the earlier ones). Third-party sizes are of whole files, headers included; bits/weight divides by the weight count. The Qwen3.8 DFloat11 row applies its scheme to those weights without the GPU-decode index its released files carry.

32 candidates from layer $L-1$, and adds exactly scored aligned candidates: the same index in every same-size source, twin neurons ($\pm$12288, an encoder flag), the sibling layer $L-16$, the v3 partners, and the unit's own rows at offsets $-1, -2, -32, \ldots, -256$, which cover RoPE pairs [35]. It then re-ranks by exact cosine and keeps the cheapest $k \leq 8$ by code length. The twin offset and the sibling distance come from Qwen3.5-27B's growth (§7); these settings, the sketch size and the candidate count were chosen in trial runs on one Qwen3.8 shard (App. B).

Results. Table 2 gives the whole-model results and Figure 1 places them. On Qwen3.8, v4 saves 144.8 MB (0.40%) over v3 and 267.7 MB over v2. Our projection from sampled layers was about 10.438, and the measurement came in 0.008 below it. Gains are larger in later shards, which hold deeper layers (Figure 2): about −0.02 in shards 1–2 and −0.04 to −0.08 in shards 12–17, though

not monotonically. A likely reason, not tested, is that deeper layers have an $L - 16$ sibling and more earlier tensors to draw from. By type, v4 gains most on full-attention q_proj (-0.137 per weight) and on linear-attention qkv, z and out (-0.059 to -0.071). For q, in the research search at layer 35, 91% of q rows' best partners were the unit's own earlier rows (RoPE pairs, other heads and the output-gate rows' shared direction). MLP gate, up and down gain -0.024 to -0.046 over v3. Embeddings, lm_head, norms and the vision tower are coded plain. Table 6 in the appendix has the breakdown. On Qwen3.6-27B, with the same encoder settings, match mode saves 0.0757 bits/weight over plain coding (Qwen3.8: 0.0771 over v2). It covers 503 of 1,199 tensors, and its largest gains there are on down_proj (-0.146 per weight against plain) and on linear-attention out, q_proj and o_proj (-0.14 to -0.15). The two models share a lineage (§7), so Qwen3.6-27B does not test other families.

Qwen2.5-32B-Instruct is outside that lineage. With the encoder settings unchanged (twin offset and sibling distance included), match saves 0.1915 bits/weight over plain coding, more than on either 27B model, and covers 437 of 771 tensors. Each MLP type gains 0.19–0.23 per weight against plain (Qwen3.6-27B: 0.03–0.15), q_proj and k_proj 0.16–0.18, o_proj 0.11 and v_proj 0.07 (Table 8). Per shard, v4 is 10.04–10.48 bits/weight. Its plain-only size, 10.5735, is also higher than either 27B model's. We did not measure its order-0 entropy, so we make no below-order-0 claim for it. DFloat11's paper reports this model at 45.53 GB, 69.48% of BF16 (11.12 bits/weight; its v1 table); v4 stores it in 64.89%. We did not re-run DFloat11. Its v3 table drops the model and lists QwQ-32B, of the same architecture and size, at 10.90 against v1's 11.12, so the v1 figure may not match DFloat11's current format. This is one model of one more generation, not a test across families.

Generic compressors sit well above the order-0 entropy: zstd -3 at 12.4590 over the whole model and, on the three shards, zstd -19 at 12.1200 and xz -9 at 11.2028, against v4's 10.4174 on the same shards. ZipNN, which Huffman-codes the exponent byte, measures 10.6155, close to the DFloat11 scheme. DFloat11's released files carry more than its scheme: for its Qwen3-8B and Qwen3-14B releases, the safetensors headers give 8 raw bits, a Huffman exponent of 2.684–2.694, and 0.21 bits/weight of GPU-decode index, for 10.897–10.907 in all. Of the $\approx$0.29 between those released files and our 10.61, about 0.21 is that index, which buys GPU-parallel decode Ingot does not have, and about 0.08 is the model (Qwen3's Huffman-coded exponent costs more than Qwen3.8's 2.61).

How much is growth-specific. The twin offset and the sibling layer exist only because Qwen3.5-27B was grown. At layer 36, excluding twin and sibling partners lowers the best-partner gain on gate from $+0.033$ to $+0.024$ and on up from $+0.025$ to $+0.020$ per weight; linear-attention tensors lose 4–11% of their gain. Our earlier projection of the sibling and twin levers was $\approx$0.0085 bits/weight model-wide (an estimate, not a measurement). On Qwen2.5-32B-Instruct, which we did not test for growth and whose shapes these settings were not chosen for, match gains 0.1915 bits/weight; how that gain splits between twin, sibling and other partners is not measured.

Speed. v4 encoding is dominated by the match search. Qwen3.8 took 17,319 s to encode and 4,683 s to decode on a dual Xeon E5440 server (8 cores, SSE4.1), reading the source over a network file system at about 70–85 MB/s: 3.2 and 11.9 MB/s of source. Shards 14–18 ran on 3 threads at low priority on a shared machine. Qwen3.6-27B ran on the same machine at 3 threads, shared with other jobs: 34,872 s to encode and 3,532 s to decode. Qwen2.5-32B-Instruct (65.5 GB) ran the same way: 44,252 s to encode and 4,760 s to decode (1.5 and 13.8 MB/s of source). Tensors with self partners decode their blocks serially. For v3 on one 2.1 GB shard on an Apple M4 laptop (4 threads): encode 73 MB/s (1.74$\times$ slower than v2) and decode 60 MB/s on 1 thread (1.59$\times$ slower).

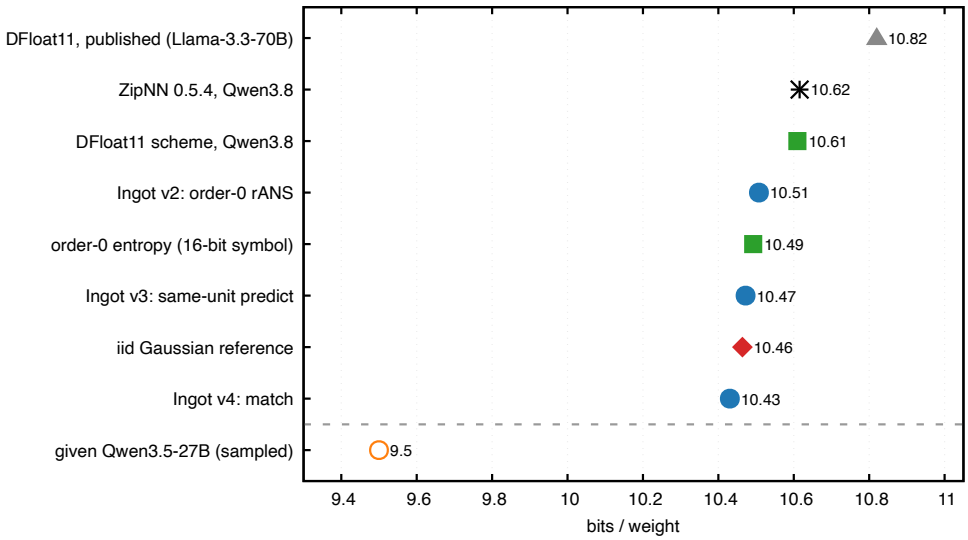

Figure 1: Bits per weight on Qwen3.8-27B (BF16 stores 16). Blue circles: codec, end to end. Green squares: exact code length computed on the whole model (order-0 and the DFloat11 scheme). Grey triangle: DFloat11's published figure on a different model. Red diamond: exact i.i.d. Gaussian reference. Black cross: ZipNN run by us on the same files (zstd −3, at 12.4590, is off the scale). Below the rule, orange: coding given Qwen3.5-27B as side information (sampled tensors weighted by type; not a codec result, App. D).

With 4 threads, decode was within 4% of v2.

v4 against v2 in practice. v4's 267.7 MB saving on Qwen3.8 costs 4,683 s of decode. Even if v2 decoded instantly, v4 shortens download plus decode only on links slower than about 57 kB/s. As a transfer format v4 does not pay on today's links; v2 is the practical exact format. On Qwen2.5-32B-Instruct the saving is larger, 784.4 MB for 4,760 s of decode, and the break-even link is about 165 kB/s, still slower than today's links. v4's value here is as a measurement: it shows end to end that the cross-tensor structure is real and codeable.

Gates. The codec test suites pass on two platforms (clang/NEON and GCC/SSE2), with identical sizes. The match-mode tests (8/8) include red tests: a corrupted coefficient fails decode, a wrong or missing reference is refused, planted cross-tensor structure codes smaller, a v3 file still decodes, and the context hash is the same at -O0 and -O2.

6 Same-unit coupling

A unit of a transformer owns one vector in several matrices, all in the residual-stream basis. For MLP neuron i these are gate row g_i, up row u_i and down column d_i. For an attention value channel they are a value row and an output column. We code d_i given the prediction $\hat{d}_i = a_i g_i + b_i u_i$, with per-unit coefficients charged at 16 bits each. That a neuron's input and output directions are related is known from interpretability work (§2); what follows measures what the relation is worth to a coder, net of its side information, and how it varies with depth and architecture.

Qwen3.8. Over seven sampled layers, down coded given its own gate and up rows saves +0.064 to +0.229 bits/weight of down. The gain peaks at layer 31 of 64. Between 87% and 96% of neurons have

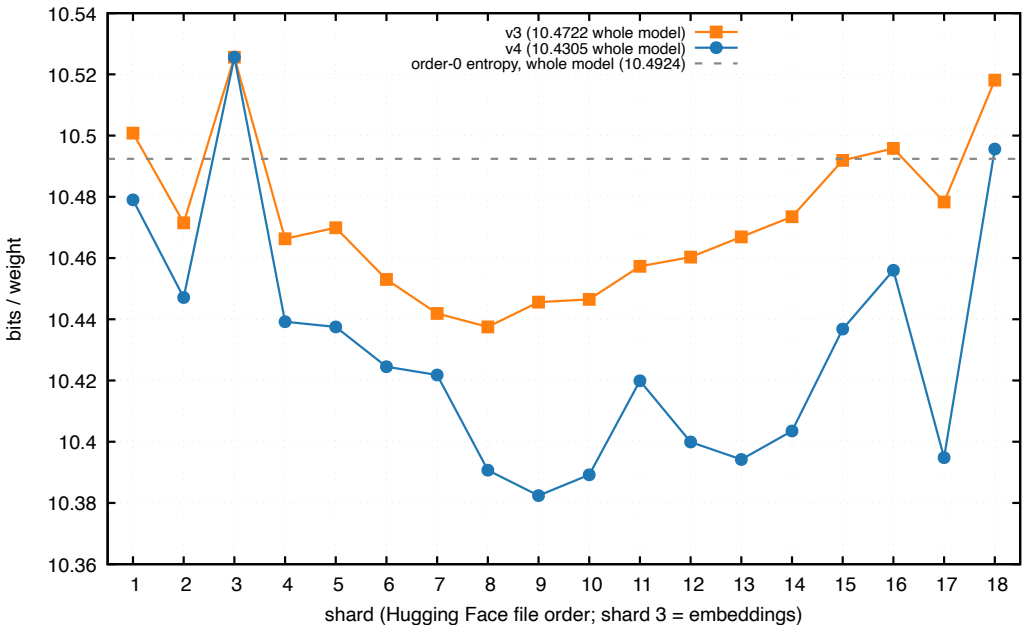

Figure 2: v3 and v4 per shard, Qwen3.8-27B. Shard 3 holds the embeddings (plain in both). A shard's v4 size is its code length given the earlier shards, which its decoder needs.

$|\rho(d_i, u_i)| > 0.1$. The shifted-partner null is at most 0.0003. Linear-attention output given value or z rows adds more. Averaged over the seven sampled layers this is 0.039 bits/weight model-wide, or 0.042 with full-attention heads (taken from one layer); both are Gaussian estimates. v3, which also uses full-attention o|v, realized 0.0354 end to end, 0.84–0.90 of them. Neither a shared map from (g, u) to d nor a per-layer map adds anything over per-neuron coefficients. The structure is per neuron, not a shared rotation.

General across families. Table 3 repeats the measurement on six more models: Llama-, OLMo-, Phi-, Mistral- and GPT-NeoX-style blocks, with gated [34] and non-gated MLPs, MHA, GQA [1] and QK-norm, stored in BF16, F16 or F32. Every model shows it, at 0.029–0.042 bits/weight model-wide (Gaussian estimate). These model-wide numbers interpolate 6–8 sampled layers linearly over depth; one layer per model was chosen in advance as a held-out check. Interpolating without it under-predicted down there in all six models (those layers sit near the peak) and over-predicted o|v in three; the model-wide number moves by at most 0.002 with or without it. Nulls sit at their floor in every layer except Phi L31 (q|k, and up|gate at +0.0005) and Mistral L1 q|k. There a direction shared by all units raised the null, and we subtracted the excess.

It follows depth. It also appears with and without gating and with every attention type we tested (Figure 3). In every model the down gain rises from layer 1 to a broad peak at 25–52% depth (+0.17 to +0.27) and falls after it. In the three 32-layer models it reaches a trough at 75–88% depth and rises again at the last layer; SmolLM2 and OLMo-2-1B keep falling to their last layer. Layers are sampled every 4–8, so the trough cannot be placed more exactly. OLMo-2 1B and 7B, trained on the same data and recipe, land in the same band, and so does Qwen3.8-27B. Pythia's non-gated GELU MLP shows it too. In SwiGLU models the coupling is mostly down–up; up given gate is near zero except in the last layer. MHA, GQA, QK-norm and hybrid linear attention all show o|v. Pythia is lowest (0.034 per block weight against 0.039–0.052), a gap within the held-out

Model	Block	Params	down	o|v	Model-wide
Pythia-1.4b	GPT-NeoX, GELU, parallel	1.4 B	0.075	0.095	0.029
SmolLM2-1.7B	Llama, SwiGLU, MHA	1.7 B	0.112	0.126	0.037
OLMo-2-1B	OLMo-2, post-norm, QK-norm	1.5 B	0.166	0.085	0.038
Phi-4-mini	fused gate_up/qkv, GQA	3.8 B	0.123	0.065	0.034
OLMo-2-7B	OLMo-2	7.3 B	0.147	0.071	0.039
Mistral-7B-v0.3	SwiGLU, GQA	7.2 B	0.119	0.087	0.041
Qwen3.8-27B†	SwiGLU, hybrid linear/full attn.	27.8 B	0.141	—	0.042

Table 3: Same-unit coupling, net bits/weight (Gaussian, coefficients charged). "down" = down given own gate and up (Pythia: given up); per-tensor columns are depth means of that tensor's net gain, model-wide numbers are sampled and interpolated over depth. †Built differently: plain means over seven sampled layers, linear-attention out|v or out|z (the better one) and full-attention heads from one layer.

interpolation error. With one old-recipe point we cannot separate recipe, GELU and the parallel block.

The sign turns negative late. Let pos be the fraction of neurons with a positive down–up relation, from the unweighted product (norm gains not folded in). In a gated MLP this sign survives the exact symmetry $(u_i, d_i) \to (-u_i, -d_i)$, so it is a property of the function, not of the parametrization; a non-gated MLP has no such symmetry. We measured pos in every layer of all seven models, on row slices (512 channels, every neuron for down; four 256-neuron blocks for up) that agree with full tensors within 0.04 (Figure 4). In all six SwiGLU models pos is high early (up to 0.97–0.99) and ends at 0.02–0.11 in the last layer, consistent with the strengthening-then-weakening pattern of [15, 16]. The final drop is sharp in Qwen3.8, Mistral and OLMo-2-7B (three to four layers from above 0.7 to below 0.25). It is gradual in the others: SmolLM2 declines from 0.90 at layer 7, Phi-4-mini drifts at 0.54–0.67 over layers 12–22, and OLMo-2-1B hovers near 0.5 (0.42–0.55) for six layers before it drops. Taking the crossing as the last downward crossing of 0.5, it sits at 67% (OLMo-2-1B), 74% (Phi-4-mini, SmolLM2), 84% (OLMo-2-7B), 87% (Mistral) and 94% (Qwen3.8) of depth; the first crossing differs only for OLMo-2-1B (53%). Pythia differs: pos is 0.98 at layer 0, mixed at layer 1, and below 0.5 from layer 2 on, consistent with the early anti-aligned neurons Gurnee et al. report for GPT-2 and Pythia-160M [17]. GELU or the parallel residual are candidate causes; we did not test them. In Qwen3.8 the neurons in the second copy block (§7) flip last.

Why (hypothesis, untested). The gradient of a neuron's down column ($\delta\, h_i$) and that of its up row ($\propto (d_i \cdot \delta)\, x$) are both rank one in the residual basis. If the residual input x and the residual error δ share directions, training pulls d_i toward $\pm u_i$. The sign would then reflect the neuron's role: building features early and clearing them late, the memory-management reading of [11]. OLMo-2's public intermediate checkpoints would allow a direct test, by tracking the coupling over training. We have not run it.

Norm gains: real but worth little. RMSNorm and LayerNorm gains track the column scales of the matrices they feed (correlation up to 0.95 in Qwen3.8, 0.99 in Mistral's first layers). But column-scale variation in trained models is small, so the gain is at most 0.0014 bits/weight model-wide, except in Mistral: 0.013–0.020 model-wide (the two figures weight the sampled layers differently), almost all of it from layers 0–1, where near-dead channels have both a tiny gain and tiny weights. The norm gain only replaces the column side information, which a scale-context coder already pays for.

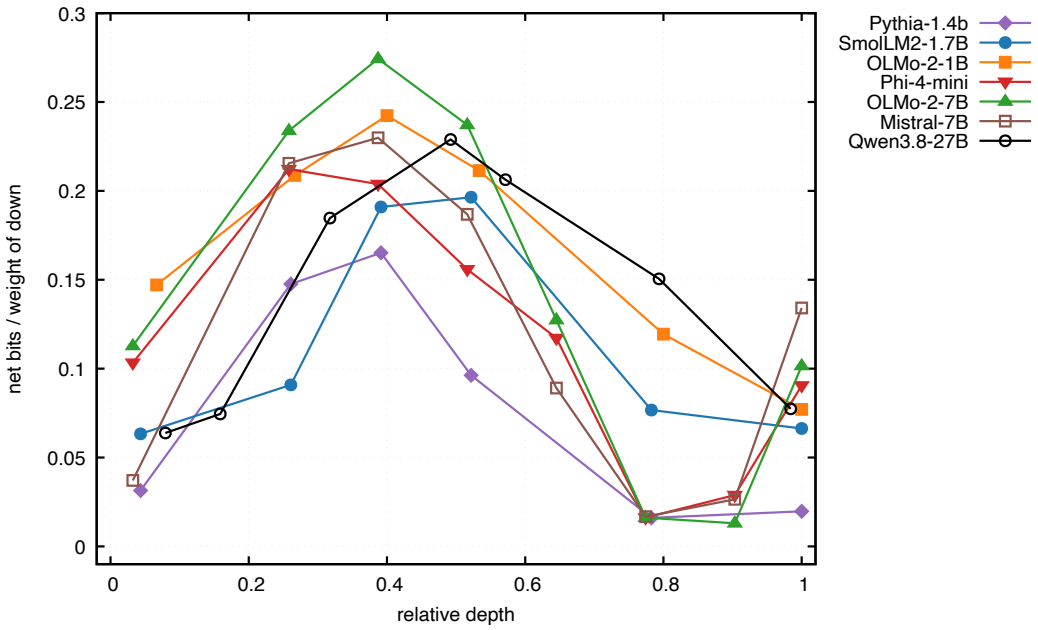

Figure 3: Net bits/weight of down coded given its own gate and up rows, by relative depth (sampled layers). All rise to a peak at 25–52% depth and fall after it; the three 32-layer models reach a trough at 75–88% and rise at the last layer.

7 Lineage and growth, read from the weights

Three index blocks. We tested whether the stored neuron order is random. On layer 36, the Spearman correlation between neuron index and the gate·up cosine has $z = +17.6$, against a null with $|z| \leq 1.1$. We then ran a changepoint search with no prior on where boundaries fall (exact dynamic programme, model order chosen by BIC). It splits every Qwen3.8 MLP layer we tested (11 of 64) into three contiguous blocks: $[0, \sim 5120)$, $[\sim 5120, 12288)$ and $[12288, 17408)$, with $|z|$ up to 60. Permuted-neuron nulls give one block, with $|z| \leq 6.4$ at a forced $K = 3$. The 12288 edge is a hard step in every layer. The ∼5120 edge is soft (4853–5928 in Qwen3.8).

The blocks are older than Qwen3.8. Qwen3.5-27B and Qwen3.6-27B have the same blocks at the same indices. Their neurons align with Qwen3.8's index for index, with no permutation (Figure 5). The same-index cosine is 0.70–0.98, against a shifted-index null of 0.003–0.011. cos(3.8, 3.6) is below cos(3.8, 3.5) at 10 of 11 layers, so Qwen3.8 descends from Qwen3.5 or a checkpoint near it rather than from 3.6. If drift multiplied independently along a chain 3.5→3.6→3.8, layer 0 (gate) would show 0.64; we measure 0.74.

Qwen3.5-27B is consistent with growth from Qwen3.5-9B. Hidden channels of two models can be compared directly through the shared vocabulary, by correlating token-embedding columns. Each 27B channel correlates with one 9B channel: 5116 of 5120 at the mapped index, mean $|r|$ 0.33–0.43, token-shuffled null 0.009. The map shows that 9B's first 1024 channels were copied to 27B channels j and $4096 + j$, and the remaining 3072 carried over once. MLP neurons follow the same pattern. 9B neurons $[0, 5120)$ appear twice, at j and $12288 + j$, and the other 7168 once. This explains the three blocks: copy A, the carried block, copy B. Same-index cosine between a 27B layer and its 9B source layer is up to 0.58, and the rotation-invariant Gram correlation up to 0.71. A

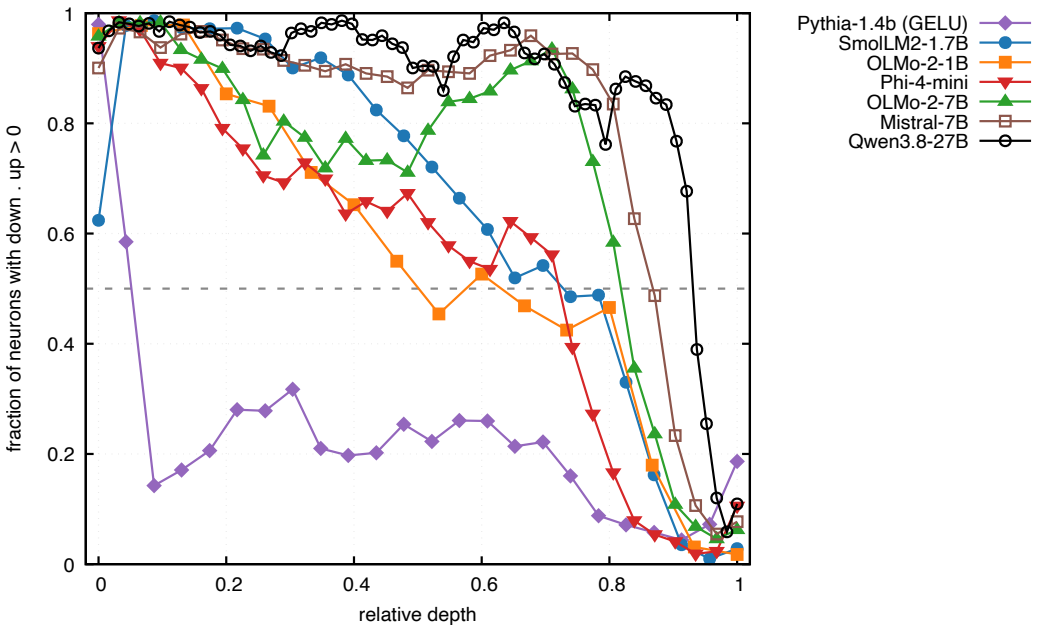

Figure 4: Fraction of MLP neurons whose down column is positively related to their up row, every layer of seven models. Shifted-partner null 0.45–0.55 (dashed: 0.5).

mismatched control pair (27B layer 0 against 9B layer 31) gives 0.000. A layer-similarity scan over all 64 layers of Qwen3.8 (256 gate rows per layer, 2,016 pairs) finds depth growth as well. 9B layers 8–23 appear three times, as layers 8–23, 24–39 and 40–55, which matches every 27B-to-9B pair above. Copies at offset 16 share cosine 0.08–0.26, while adjacent layers share 0.004 (null at most 0.013). Qwen3-8B and Qwen3-14B [48] have the same 12,288 and 17,408 MLP widths but show no link: every statistic sits at null. Twin neurons inside the 27B rule out pruning from a larger model; they do not rule out a shared earlier ancestor of the released 9B. Training since the growth step kept the layout. Inside Qwen3.8, twin neurons still correlate (mean r 0.14–0.28; null at most 0.004). Compared with the lineage tests in §2, what is new here is the growth map itself (channel, neuron and depth copies in a released family) and its use for coding.

A positive control. We ran the same tests on OLMo-2-7B's public checkpoints, whose history is known (Figure 6). Each stage-1 checkpoint is closest to its neighbours in training time, so the method recovers the training order; stage 2, base and instruct form one cluster (0.98–1.00) whose order it does not resolve. An independently trained OLMo of the same shape scores 0.000. The shift null is at most 0.0035.

For coding. The block statistics themselves are worth nothing: per-block tables and priors gain at most 0.0001, the same as their null. Per-unit scales and coefficients already carry them. The copies are worth something. On Qwen3.8, coding an MLP tensor given its sibling 16 layers earlier saves +0.003 to +0.028 per weight of that tensor (3 layers measured). Coding twin neurons given their copies saves +0.001 to +0.017 (6 layers). Both are net of all side information, and their nulls are at or below zero. Their gains add. Format v4 uses both (§5).

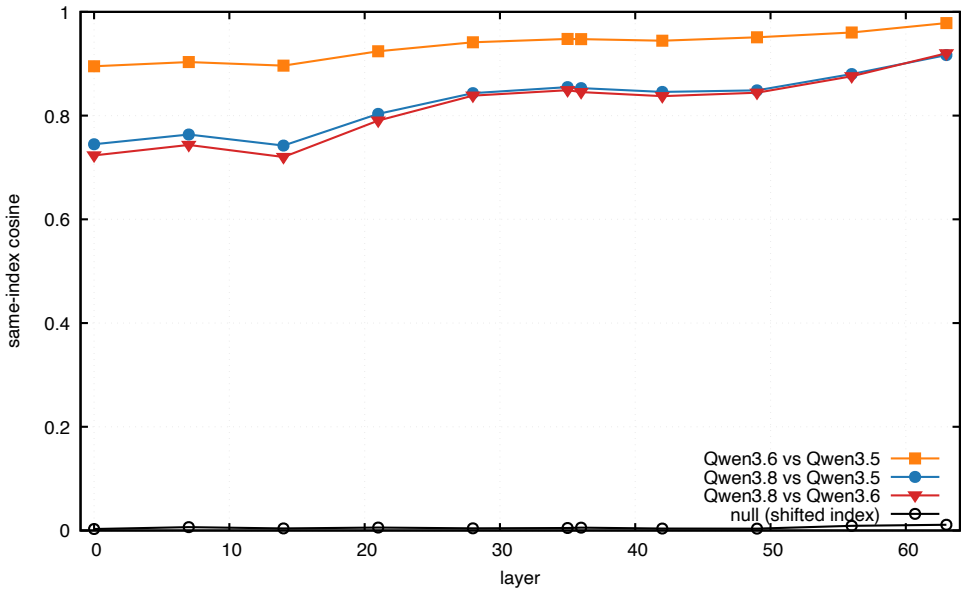

Figure 5: Same-index neuron cosine between Qwen 27B generations (mean of gate, up, down), 11 layers. Neurons keep their index across releases. Qwen3.8 is closer to 3.5 than to 3.6.

8 Limits

- **Three whole models, all dense Qwen.** Qwen3.8-27B, Qwen3.6-27B and Qwen2.5-32B-Instruct are reported end to end with v4. The first two share a lineage, and the encoder's twin and sibling settings come from it. Qwen2.5-32B-Instruct is outside that lineage, but it is one model from the same developer; no other family is coded end to end, and we have no order-0 measurement for Qwen2.5. The cross-family coupling, growth levers and side-information results are sampled (3–11 layers or single tensors) and interpolated. We label them so.

- **MoE and other architectures later.** Match mode codes 2-D tensors only, and fused 3-D expert tensors hold most of an MoE model's weights. We ran v4 end to end on Qwen3.6-35B-A3B (all shards rebuilt exact) but report no number from it: the run mostly measures plain mode. Other architectures come once the codec is final.

- **Tuned on the test model.** The encoder settings were chosen on one Qwen3.8 shard; the code length stays valid because the decoder reads every choice, but the settings may suit other models less.

- **Small gain, CPU only.** On Qwen3.8, v4 is 1.7% smaller than the DFloat11 scheme on the same weights and 0.6% below the order-0 entropy, and its decode is slow (§5): it only pays over v2 on links slower than about 57 kB/s (165 kB/s on Qwen2.5-32B-Instruct). The structure is real, but there is little of it in normally trained weights.

- **Storage format only.** Ingot today decodes back to the original file, which is then loaded. GPU decode at inference time is not finished. v4's self partners (a serial chain per tensor) and $L-16$ pools (17 decoded layers held) block per-token decode from VRAM. A device-decodable encoding (strip restarts, anchors) is being measured separately and may cost more bits. The released files stay valid v4; this paper makes no claim about inference speed or memory.

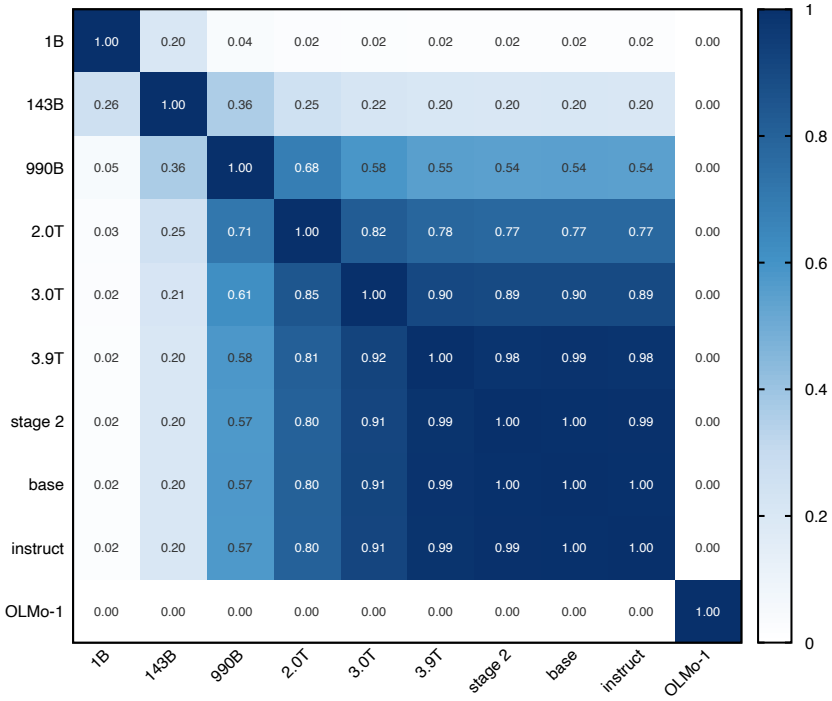

Figure 6: OLMo-2-7B layer 16, same-index cosine between checkpoints (stage-1 token counts, stage 2, base, instruct) and a separately trained OLMo-1 7B. Above the diagonal: gate rows 0–1023; below: down rows 0–255.

- **Slow encode.** The v4 match search takes hours per model on old hardware.
- **Speed not side by side.** v3 is superseded by v4. We timed v4 on whole models on one machine and v3 on one shard on another; we did not time v2, v3 and v4 side by side on one machine.
- **DFloat11's compressor was not run.** Its published 10.81–10.91 (v3) is on other models. Our 10.61 is its scheme measured on Qwen3.8's weights; its released files add a 0.21 bits/weight decode index (§5). The one same-model comparison, Qwen2.5-32B-Instruct, uses its v1 figure (11.12), which we did not re-measure; its v3 table gives a same-architecture model 0.22 less.
- **One model behind the near-Gaussian picture.** Order-0 and kurtosis are reported for Qwen3.8 only; Qwen3.6-35B-A3B's attention tensors sit 0.1–0.2 higher.
- **Open causes.** We did not test why the coupling exists, why its sign turns, or why Pythia differs.
- **Work in progress.** A token-dictionary predictor (neurons predicted from a few embedding or output-head rows) is being tested for a later format version and is not part of this paper.
- **Literature.** Our search for prior work was a set of queries, not a survey.

9 Reproducibility

Code. The codec (format, encoder, decoder, CLI, tests) is open source in Foundry, `https://github.com/dcherrera/Foundry`, directory `ingot/`, at commit `64b1939f272d`. The encoder that produced the Qwen3.8 and Qwen3.6 results has the same `ingot/` sources as that commit; the Qwen2.5 run used the same encoder build directory with the same flags. Build and test with a C compiler and `make`:

```
git clone https://github.com/dcherrera/Foundry && cd Foundry
git checkout 64b1939f272d
make -C ingot build/ingot && make -C ingot test
```

Encode. Shards are encoded in name order; shard k gets the earlier source shards as references. The non-default flags are the twin offset and the sibling search:

```
ingot encode -j 8 --refs <shard 1> ... --refs <shard k-1> \
    --twin-offset 12288 --search-layers 1 <shard k>.safetensors <shard k>.ingot
```

The script that runs the whole chain is

```
ingot/scripts/ingot_measure_chain.sh -j 8 <model-dir> <sha256-file> <work-dir> \
    -- --twin-offset 12288 --search-layers 1
```

It encodes each shard, decodes the written file with the earlier *rebuilt* shards as references, and checks the rebuilt SHA-256 against the Hugging Face hash. Its total line for Qwen3.8 is `output_bytes 36221798920`.

Decode and verify. Decode in name order, passing the shards already decoded:

```
ingot decode --refs <decoded 1> ... --refs <decoded k-1> \
    <shard k>.ingot <shard k>.safetensors
ingot verify --refs ... <shard k>.ingot [<original>]
```

`verify` checks every tensor hash and the file's SHA-256 without writing.

Models and files. The whole models, by Hugging Face repository and revision, are Qwen/Qwen3.8-27B `37561816e66b`, Qwen/Qwen3.6-27B `8f33b98937c6` and Qwen/Qwen2.5-32B-Instruct `5ede1c97bbab`; App. H lists the other models. The Qwen3.8-27B v4 files are on Hugging Face as `TeamIDE/Qwen3.8-27B-Ingot` (18 files, 36,221,798,920 bytes, with SHA-256 sums of the `.ingot` files and the original shards).

Data. The data behind every figure and table (per-shard codec logs, per-tensor sizes, the cross-family and per-layer coupling tables, sign, lineage and checkpoint matrices, lever measurements, the baseline runs, and the gnuplot scripts that draw the figures from them) is in the paper's data release (`release/README.md` lists each file and the figure or table it backs). The research measurement tools behind the sampled numbers are described here, not released.

10 Conclusion

Inside one trained BF16 tensor we found about 0.014 bits/weight of row and column scale structure, row-to-row structure in attention projections, and in MLP tensors almost nothing else. Most of what is left lives between tensors of the same model: a neuron's input and output vectors, its copies from growth, and its sibling layers. A codec built on these sources, with no other model needed, stores Qwen3.8-27B in 10.4305 bits/weight, Qwen3.6-27B in 10.4328 and Qwen2.5-32B-Instruct in 10.3820, bit for bit. The saving over simple exact coding is small. The measurements also give a coding value for a known input–output alignment of neurons across six families, with its sign turning late in every SwiGLU model, and show that a family's growth can be read from its weights.

Acknowledgments

Writing, experiment code and analysis were done with AI assistance (Anthropic's Claude) under the author's direction. All results are measurements on real models and hardware; the author verified them and takes full responsibility for the paper.

References

[1] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[2] Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, et al. SmolLM2: When smol goes big – data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.

[3] Stella Biderman et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, 2023.

[4] Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. Net2Net: Accelerating learning via knowledge transfer. In *International Conference on Learning Representations*, 2016.

[5] John H. Conway and Neil J. A. Sloane. *Sphere Packings, Lattices and Groups*. Springer, 3 edition, 1999.

[6] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *arXiv preprint arXiv:2305.14314*, 2023.

[7] Tim Dettmers et al. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. *arXiv preprint arXiv:2306.03078*, 2023.

[8] Wenyu Du, Tongxu Luo, Zihan Qiu, Zeyu Huang, Yikang Shen, Reynold Cheng, Yike Guo, and Jie Fu. Stacking your transformers: A closer look at model growth for efficient LLM pre-training. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2405.15319.

[9] Jarek Duda. Asymmetric numeral systems: Entropy coding combining speed of Huffman coding with compression rate of arithmetic coding. *arXiv preprint arXiv:1311.2540*, 2013.

[10] Vage Egiazarian et al. Extreme compression of large language models via additive quantization. *arXiv preprint arXiv:2401.06118*, 2024. ICML 2024.

[11] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, et al. A mathematical framework for transformer circuits. Transformer Circuits Thread, `https://transformer-circuits.pub/2021/framework/`, 2021.

[12] Ruibo Fan, Xiangrui Yu, Xinglin Pan, Zeyu Li, Weile Luo, Qiang Wang, Wei Wang, and Xiaowen Chu. ZipServ: Fast and memory-efficient LLM inference with hardware-aware lossless compression. In *ASPLOS*, 2026. arXiv:2603.17435.

[13] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022. ICLR 2023.

[14] Georgi Gerganov and contributors. ggml / llama.cpp quantization formats (Q8_0, K-quants). `https://github.com/ggml-org/llama.cpp`, 2023.

[15] Sebastian Gerstner and Hinrich Schütze. Understanding gated neurons in transformers from their input-output functionality. *arXiv preprint arXiv:2505.17936*, 2025.

[16] Sebastian Gerstner, Hilal AlQuabeh, Kentaro Inui, and Hinrich Schütze. Weakening neurons: An input-output functionality in transformers with outsize influence. In *Proceedings of EMNLP*, 2026. arXiv:2609.18612.

[17] Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. Universal neurons in GPT2 language models. *arXiv preprint arXiv:2401.12181*, 2024.

[18] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *International Conference on Learning Representations*, 2016. arXiv:1510.00149.

[19] Yongchang Hao, Yanshuai Cao, and Lili Mou. NeuZip: Memory-efficient training and inference with dynamic compression of neural networks. *arXiv preprint arXiv:2410.20650*, 2024.

[20] Anat Heilper and Doron Singer. Lossless compression of neural network components: Weights, checkpoints, and K/V caches in low-precision formats. *arXiv preprint arXiv:2508.19263*, 2025.

[21] Moshik Hershcovitch, Leshem Choshen, Andrew Wood, Ilias Enmouri, Peter Chin, Swaminathan Sundararaman, and Danny Harnik. Lossless and near-lossless compression for foundation models. *arXiv preprint arXiv:2404.15198*, 2024.

[22] Moshik Hershcovitch, Andrew Wood, Leshem Choshen, Guy Girmonsky, Roy Leibovitz, Ilias Ennmouri, Michal Malka, Peter Chin, Swaminathan Sundararaman, and Danny Harnik. ZipNN: Lossless compression for AI models. *arXiv preprint arXiv:2411.05239*, 2024. IEEE CLOUD.

[23] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

[24] Dhiraj Kalamkar et al. A study of BFLOAT16 for deep learning training. *arXiv preprint arXiv:1905.12322*, 2019.

[25] Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, et al. SOLAR 10.7B: Scaling large language models with simple yet effective depth up-scaling. In *NAACL (Industry Track)*, 2024. arXiv:2312.15166.

[26] Yuriy Kim and Evgeny Belyaev. An efficient compression of deep neural network checkpoints based on prediction and context modeling. In *IEEE NW Russia Young Researchers in Electrical and Electronic Engineering Conference*, 2025. arXiv:2506.12000.

[27] Ismail Lamaakal. Motion-compensated weight compression. *arXiv preprint arXiv:2605.24754*, 2026.

[28] Tingfeng Lan, Zirui Wang, Yunjia Zheng, Zhaoyuan Su, Juncheng Yang, and Yue Cheng. TensorDex: A compact, tensor-centric storage system for modern AI models. *arXiv preprint arXiv:2604.17104*, 2026.

[29] Olivier Ledoit and Michael Wolf. Analytical nonlinear shrinkage of large-dimensional covariance matrices. *The Annals of Statistics*, 48(5):3043–3065, 2020.

[30] Kang-Ho Lee, JoonHyun Jeong, and Sung-Ho Bae. An inter-layer weight prediction and quantization for deep neural networks based on a smoothly varying weight hypothesis. *arXiv preprint arXiv:1907.06835*, 2019.

[31] Siyuan Li, Haoxuan Zeng, Xin Luo, Fernando Jia, Florence Li, Zhengyang Geng, Zico Kolter, Tai Sing Lee, and Tianqin Li. Witness overlap: Directional provenance inside open-weight model families. In *Advances in Neural Information Processing Systems*, 2026. arXiv:2609.31784.

[32] Microsoft. Phi-4-Mini technical report: Compact yet powerful multimodal language models via mixture-of-LoRAs. *arXiv preprint arXiv:2503.01743*, 2025.

[33] Wanyi Ning, Jingyu Wang, Qi Qi, Mengde Zhu, Haifeng Sun, Daixuan Cheng, Jianxin Liao, and Ce Zhang. FM-Delta: Lossless compression for storing massive fine-tuned foundation models. In *Advances in Neural Information Processing Systems*, volume 37, pages 66796–66825, 2024.

[34] Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[35] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[36] Hongshi Tan, Yao Chen, Gustavo Alonso, Weng-Fai Wong, and Bingsheng He. Approaching Shannon bound with lossless LLM weight compression. In *International Symposium on Computer Architecture (ISCA)*, 2026. arXiv:2606.15789.

[37] Team OLMo. 2 OLMo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024. COLM 2025.

[38] James Townsend, Thomas Bird, and David Barber. Practical lossless compression with latent variables using bits back coding. In *International Conference on Learning Representations*, 2019.

[39] Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. QuIP#: Even better LLM quantization with Hadamard incoherence and lattice codebooks. *arXiv preprint arXiv:2402.04396*, 2024. ICML 2024.

[40] Elena Voita, Javier Ferrando, and Christoforos Nalmpantis. Neurons in large language models: Dead, n-gram, positional. *arXiv preprint arXiv:2309.04827*, 2023.

[41] Suqing Wang, Ziyang Ma, Li Xinyi, and Zuchao Li. Ghost in the transformer: Detecting model reuse with invariant spectral signatures. In *AAAI*, 2026. arXiv:2511.06390.

[42] Zirui Wang, Tingfeng Lan, Zhaoyuan Su, Juncheng Yang, and Yue Cheng. ZipLLM: Efficient LLM storage via model-aware synergistic data deduplication and compression. *arXiv preprint arXiv:2505.06252*, 2025.

[43] Simon Wiedemann, Heiner Kirchhoffer, Stefan Matlage, Paul Haase, Arturo Marban, Talmaj Marinc, David Neumann, Ahmed Osman, Detlev Marpe, Heiko Schwarz, Thomas Wiegand, and Wojciech Samek. DeepCABAC: Context-adaptive binary arithmetic coding for deep neural network compression. *arXiv preprint arXiv:1905.08318*, 2019.

[44] Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. LLaMA Pro: Progressive LLaMA with block expansion. In *ACL*, 2024. arXiv:2401.02415.

[45] Rui Xie, Asad Ul Haq, Yunhua Fang, Linsen Ma, Zirak Burzin Engineer, Liu Liu, and Tong Zhang. TRACE: Unlocking effective CXL bandwidth via lossless compression and precision scaling. *arXiv preprint arXiv:2509.03377*, 2025.

[46] Xianglong Yan, Chengzhu Bao, Zhiteng Li, Tianao Zhang, Shaoqiu Zhang, Ruobing Xie, Xingwu Sun, and Yulun Zhang. D^2Quant: Accurate low-bit post-training weight quantization for LLMs. *arXiv preprint arXiv:2602.02546*, 2026.

[47] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[48] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[49] Jinwu Yang, Jiaan Wu, Zedong Liu, Xinyang Ma, Hairui Zhao, Yida Gu, et al. ENEC: A lossless AI model compression method enabling fast inference on Ascend NPUs. In *International Symposium on Computer Architecture (ISCA)*, 2026. arXiv:2604.03298.

[50] Zeyu Yang, Tianyi Zhang, Jianwen Xie, Chuan Li, Zhaozhuo Xu, and Anshumali Shrivastava. To compress or not? Pushing the frontier of lossless GenAI model weights compression with exponent concentration. *arXiv preprint arXiv:2510.02676*, 2025.

[51] Patrick Yubeaton, Tareq Mahmoud, Shehab Naga, Pooria Taheri, Tianhua Xia, Arun George, Yasmein Khalil, Sai Qian Zhang, Siddharth Joshi, Chinmay Hegde, and Siddharth Garg. Huff-LLM: End-to-end lossless compression for efficient LLM inference. *arXiv preprint arXiv:2502.00922*, 2025.

[52] Boyi Zeng, Lizheng Wang, Yuncong Hu, Yi Xu, Chenghu Zhou, Xinbing Wang, Yu Yu, and Zhouhan Lin. HuRef: HUman-REadable fingerprint for large language models. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2312.04828.

[53] Tianyi Zhang, Mohsen Hariri, Shaochen Zhong, Vipin Chaudhary, Yang Sui, Xia Hu, and Anshumali Shrivastava. 70% size, 100% accuracy: Lossless LLM compression for efficient GPU inference via dynamic-length float (DFloat11). In *Advances in Neural Information Processing Systems*, 2025. arXiv:2504.11651v3.

[54] Sally Zhu, Ahmed Ahmed, Rohith Kuditipudi, and Percy Liang. Independence tests for language models. *arXiv preprint arXiv:2502.12292*, 2025.

A Levers that did and did not pay

We tried every lever in code even when an estimate said it would fail; estimates only set the order. Table 4 lists the levers that did not pay and Table 5 those that did.

A ceiling that was not one. Before v4, we estimated how much any predictor could gain from a fixed feature set. A small network per tensor role, trained on 12 layers and tested on 7, saw unit scales, neighbours, same-unit partners and per-unit fits. It reached +0.052 bits/weight model-wide (sampled; null +0.006). Taken from v2's 10.508, that implied about 10.456. v4 measures 10.4305, below that "ceiling", because it uses sources the feature set did not have: the previous layer, the sibling layer and the unit's own earlier rows. A ceiling bounds only the information it is given.

Lever	Result (net, all side information charged)
Calibration data as side info	A shipped set cannot pay (savings $\leq I(W;C) \leq H(C)$). A free public corpus: the mean gradient is reliable (split-half r 0.65–0.99) yet uncorrelated with the weights, $\lvert\rho\rvert \leq 0.015$ (≤ 0.019 extrapolated to infinite data). A K-FAC step on the last layer reaches ρ 0.05–0.07 (0.002–0.004 bits on that tensor) and grows with data; middle layers stay near 0, so we project $\leq$0.0002 model-wide (Qwen3-0.6B).
Training forensics in low bits	Contexts that a training artefact would leave in the low mantissa bits (column parity, neighbours' low bits, the LSB): every one $\leq$0.00002; LSB $P(1) = 0.5001$. Planted twins are found (+0.31, +0.46).
Low-rank predictor	Net loss at every rank 8–256: factors cost more than they save.
Tensor networks	Folding a matrix into a quantized tensor train (row and column index bits interleaved): every fold's singular-value spectrum equals that of shuffled values; only the plain matrix cut sits 0.1–0.4 bits below (mild low rank, see above).
Weights as an image	Neighbours carry nothing in stored order; after a charged reorder, 0.013 on one of four tensors and $\leq$0.004 on the others; 2-D image predictors (MED) lose.
Layer-to-layer deltas	Same index, stored order: rel. RMS 1.411 ($\approx \sqrt{2}$, unrelated); coding given $L-1$ at the same index costs more. Aligned partners in $L-1$ do pay (Table 5).
MoE expert deltas	$\lvert\cos\rvert$ 0.01–0.03; row match at chance even up to permutation; trees lose 0.03–0.17.
Lattice shaping (E8, Leech [5])	Lossless lattice coders lose 0.016–0.13. Total correlation of 8-blocks of magnitude class, sign and LSB is $\leq$0.0003 above null.
Pseudo-random structured masks	Masks from a seed (hashed Bernoulli, expander, circulant, block, butterfly; 6 families $\times$ 64 seeds) as a context: mutual information with the weights sits at the plug-in bias, and every net is ≤ 0.
Bits-back / shuffle coding	Neuron order is worth at most 0.00082 per MLP weight even if dropped; exact files keep it. Prototype mixtures lose after prototype cost. BB-ANS [38] round trip exact.
Canonical gauge forms	Exact rescaling and sign moves of up row and down column (and a sort by gate RMS) to a canonical form: net ≈ 0.
Integer wavelets, block matching	Lose 0.02–2.0; -0.0002.
Adaptive binary context mixing	Loses to static tables (weight distributions are stationary).
Exact repeats	Exact repeats of weight runs (n = 2–16) at the shuffled-control rate, in a 22% sample of the weights; no duplicate rows.
BF16 as Q8_0 + residual	Residual given the Q8_0 codes 2.64 bits/weight, pure rounding noise. Q8_0 is a function of the weights, so $H(W) = H(Q) + H(W \mid Q)$: the split saves nothing.
A tensor as one long integer	No pattern (App. C).

Table 4: Levers that did not pay, Qwen3.8-27B unless noted. Numbers are per weight of the tensor named unless marked model-wide.

B Encoder trials

The v4 encoder settings were chosen on Qwen3.8 shard 11 (layers 37–39) with the earlier shards as references. Two search bugs mattered. First, the ± 1 projection used a linear xorshift, whose

Lever	Result (net saving, all side information charged)
Real, used in v3/v4	
Same-unit coupling	§6.
Best partner over earlier tensors	Layer 36: gate +0.033, up +0.025, linear q/k/v +0.065 to +0.086, z +0.079; nulls ≤ -0.0004.
Siblings, twins, RoPE pairs	§7; q second-of-pair rows +0.036 to +0.104.
Real, not in the codec	
Cross-layer covariance prior	Gate coded with the previous layer's gate+up covariance (shrinkage [29]): +0.026 to +0.062 over a diagonal prior on 5 layers (layer 36: +0.051); needs ∼2,560 MACs per weight and a deterministic eigen/Cholesky step, so not in v4.
Embedding / lm_head via BPE merge tree	+0.011 / +0.065 per weight of that table (head: ids $< 65{,}536$ only).
Row × column scale context	0.003–0.025 per tensor for ∼30% slower decode; dropped.

Table 5: Levers that are real, Qwen3.8-27B. Numbers are per weight of the tensor named.

low sketch dimensions were periodic in the channel index; synthetic i.i.d. tests could not see this. Second, at 128 dims the sketch noise maximum (∼0.4 over $\sim 10^5$ candidates) swamps true partners at $|\cos| \approx 0.15$. Exactly scored aligned candidates fixed the sibling, twin and RoPE cases: on that shard the share of gate units with a partner rose from 4% to 93%. Moving to 512 dims with 32 candidates gained a further 0.009 on the shard.

C A tensor as one long integer

We read layer-36 down and gate runs of Qwen3.8 (1.1–1.3 M consecutive weights) as one integer N and tested whether arithmetic on it exposes structure. Multiplying by $\lfloor \pi 2^{L-2} \rfloor$ is a bijection, so it cannot help: the product codes +21.48 bits/weight worse, as an i.i.d. Gaussian control does (+21.50). Residues of 64-bit chunks mod 3–61, 251–257, 4093, 65521 and $2^{16} \pm 1$, against within-row shuffles, gain at most +0.0010 (the Gaussian control reaches +0.0005); small primes gain at most 0.000011. No pattern.

D With side information

Every Qwen3.5-generation model we sampled, and every fine-tune of one, descends from Qwen3.5 weights. With a related model available to the decoder, coding is much cheaper. These numbers are sampled code lengths from research coders, not end-to-end codec runs.

- Qwen3.8-27B given Qwen3.5-27B costs about 9.50 bits/weight with a context-table coder (samples weighted by layer; 9.57 weighted by elements). A step coder gives 9.59, and the public codec's v1 delta mode 9.89. Learned maps (affine, per row or column, nonlinear, low rank, full linear) add only 0.017 over plain subtraction. The residual behaves like near-Gaussian independent drift.
- Close fine-tunes cost far less. SABER, a Qwen3.6-35B-A3B tune whose experts are 99.94% bit-identical to its base, costs 0.13 bits/weight (experts 0.006; read from its F16 GGUF, since its safetensors repository is gated). Ornith-1.5-9B given Qwen3.5-9B costs 5.9–6.9, and Nex-N2-mini given Qwen3.5-35B-A3B-Base 6.84 (experts).

- Five released models stored as a tree take 137.7 GB, against 190.5 GB at order-0 per model (−27.7%; sampled step-coder edge costs). The saving is SABER and Nex-N2-mini coded given Qwen3.6; Ornith and Qwen3.8 stay alone. Hidden common bases never pay.

E Quantized formats, and a lossy use

Exact Q8_0 and K-quants. The same approach applies to the integer codes of GGUF files [14]. For Q8_0 we code the block scale as the BF16 maximum it was quantized from, recomputed in integers, and the position and sign of the implied ±127 code. A scale that is not reachable that way (a file not quantized from BF16) takes an escape: in Qwen3-0.6B's own GGUF, 88% of layer scales did. The other codes go in 8 scale contexts. The K-quant contexts were chosen on Qwen3-1.7B, one of the test files below. On real files:

- a Q8_0 GGUF of Ornith-1.5-9B: −7.53% (8.5 to 7.859 bits/weight);
- Qwen3-1.7B Q4_K_M and Q3_K_M: −5.92% and −5.50%;
- Qwen3.6-35B-A3B Q3_K_M: −6.26% (17.12 to 16.05 GB).

The first three were written to disk and rebuilt byte-identical, with SHA-256 equal to the source file's (Hugging Face's for the Qwen3-1.7B files). The Qwen3.6 file was coded in memory only (not written, for disk space): every tensor was decoded back and compared, and the source stream's SHA-256 matched. Single-thread decode runs at 138 (Q8_0) to 278 (Q3_K) million weights/s on an M-series laptop. The GGUF files are not pinned to revisions in our records, and the logs of these runs were not kept, so these sizes are not in the data release.

Lossy quantization. The coupling also helps a quantizer. Predict down from its own *quantized* gate and up rows, then quantize only the residual. On Qwen3-0.6B, at equal or higher baseline bits with round-to-nearest, KL to BF16 is ×0.868 the baseline's at 4 bits (95% interval 0.82–0.91) and ×0.912 at 3 bits (0.82–1.02, not significant at 5%). These are fits across 5–7 group geometries. On top of GPTQ the ratio is ×0.92, from a single bit-matched pair per width, while group geometry alone moves 3-bit KL between 0.97 and 1.29 (a factor of about 1.3), so the GPTQ figure is weak evidence. The cost is 0.031 bits/weight on down. This is one small model and one evaluation text, with no shuffled-partner arm on the model (only in the tool's red test).

F Codec v4 per tensor type

G Same-unit coupling per layer

Net bits/weight of the predicted tensor (Gaussian, coefficients charged); pos = fraction of neurons with a positive down–up relation, from full tensors where recorded (Figure 4 uses row slices).

H Models and revisions

Type	Weights (G)	v3	v4	Δ type	Δ model
mlp.gate_proj	5.70	10.4981	10.4525	−0.0456	−0.00936
linear_attn.in_proj_qkv	2.52	10.5102	10.4389	−0.0713	−0.00646
mlp.up_proj	5.70	10.4914	10.4642	−0.0273	−0.00560
self_attn.q_proj	1.01	10.5249	10.3875	−0.1374	−0.00498
mlp.down_proj	5.70	10.3753	10.3517	−0.0236	−0.00484
linear_attn.in_proj_z	1.51	10.4826	10.4200	−0.0627	−0.00341
linear_attn.out_proj	1.51	10.4230	10.3642	−0.0588	−0.00320
self_attn.o_proj	0.50	10.4676	10.3964	−0.0712	−0.00129
k, v, a, b, mtp.*	0.62			−0.03 to −0.29	−0.00260
embed, lm_head, norms, vision	3.0			0	0

Table 6: Qwen3.8-27B, whole model, v3 to v4. The type changes sum to the file-level change (−0.0417).

Type	Weights (G)	Plain	v4	Δ type	Δ model
mlp.down_proj	5.79	10.5055	10.3597	−0.1458	−0.03040
mlp.gate_proj	5.79	10.4992	10.4538	−0.0454	−0.00946
linear_attn.out_proj	1.51	10.5168	10.3633	−0.1535	−0.00834
linear_attn.in_proj_qkv	2.52	10.5113	10.4381	−0.0732	−0.00663
mlp.up_proj	5.79	10.4968	10.4657	−0.0311	−0.00649
self_attn.q_proj	1.07	10.5334	10.3872	−0.1462	−0.00563
linear_attn.in_proj_z	1.51	10.5058	10.4189	−0.0868	−0.00472
self_attn.o_proj	0.53	10.5337	10.3932	−0.1405	−0.00271
k, v, a, b	0.20			−0.14 to −0.19	−0.00128
embed, lm_head, norms, vision	3.06			0	0

Table 7: Qwen3.6-27B, whole model, plain coding to v4 (no v3 run). The MTP tensors are counted in their type rows here, not in a separate row as in Table 6. The type changes sum to −0.0757, the file-level change.

Type	Weights (G)	Plain	v4	Δ type	Δ model
mlp.gate_proj	9.06	10.5880	10.3534	−0.2346	−0.06488
mlp.up_proj	9.06	10.5668	10.3610	−0.2057	−0.05689
mlp.down_proj	9.06	10.5616	10.3683	−0.1933	−0.05346
self_attn.q_proj	1.68	10.5872	10.4273	−0.1599	−0.00819
self_attn.o_proj	1.68	10.5230	10.4164	−0.1067	−0.00546
self_attn.k_proj	0.34	10.6150	10.4308	−0.1842	−0.00189
self_attn.v_proj	0.34	10.5588	10.4840	−0.0748	−0.00077
embed, lm_head, norms, biases	1.56			0	0

Table 8: Qwen2.5-32B-Instruct, whole model, plain coding to v4. The type changes sum to −0.1915, the file-level change.

Model	Layer	down|gate,up	up|gate	o|v	q|k	pos
OLMo-2-7B	1	0.113	−0.000	0.018	0.054	0.99
	8	0.234	0.007	0.079	0.042	0.72
	12 (held out)	0.274	0.009	0.061	0.038	0.78
	16	0.237	0.009	0.116	0.023	0.78
	20	0.127	0.007	0.112	0.040	0.91
	24	0.016	0.002	0.032	0.026	0.77
	28	0.013	0.005	0.059	0.031	0.07
	31	0.102	0.036	0.127	0.046	0.06
Mistral-7B-v0.3	1	0.037	0.006	0.002	0.024*	
	8	0.216	0.011	0.124	0.047	
	12 (held out)	0.230	0.007	0.061	0.039	0.90
	16	0.187	0.011	0.124	0.007	
	20	0.089	0.005	0.115	0.005	0.93
	24	0.017	0.003	0.045	0.001	0.91
	28	0.026	0.010	0.113	0.007	0.23
	31	0.134	0.042	0.154	0.010	0.07
Qwen3.8-27B	5	0.064	0.012			
	10	0.075	0.011			0.97
	20	0.185	0.009			
	31 (full attn.)	0.229	0.003			0.90
	36	0.206	0.002			
	50	0.151	0.007			0.78
	62	0.077	0.020			

Table 9: Per-layer coupling for three models (*: null-adjusted). All seven models' per-layer tables, nulls and held-out checks are in the data release. Nulls: OLMo-2-7B and Mistral at the coefficient floor (≤ -0.003) except Mistral L1 q|k (+0.005, subtracted); Qwen3.8 shifted-partner null ≤ 0.0003.

Hugging Face repository	Revision	Used for
Qwen/Qwen3.8-27B	37561816e66b	codec, baselines, all Qwen3.8 studies
Qwen/Qwen3.6-27B	8f33b98937c6	codec
Qwen/Qwen2.5-32B-Instruct	5ede1c97bbab	codec
Qwen/Qwen3.6-27B	1cda14dd71a3	lineage (same weights as 8f33b98937c6)
Qwen/Qwen3.5-27B	81a287ca484a	lineage, side information
Qwen/Qwen3.5-9B	c20223623576	growth source, side information
Qwen/Qwen3-8B-Base	49e3418fbbbc	lineage control
Qwen/Qwen3-14B-Base	0b0bd3732e2c	lineage control
Qwen/Qwen3.5-0.8B	2fc06364715b	embedding control
Qwen/Qwen3.5-2B	15852e8c1636	embedding control
Qwen/Qwen3.5-4B	851bf6e806ef	embedding control
Qwen/Qwen3.5-35B-A3B	2508b2437e38	embedding control
Qwen/Qwen3.6-35B-A3B	345d467b117d	side information
Qwen/Qwen3.5-35B-A3B-Base	0f0813072d23	side information
EleutherAI/pythia-1.4b	fedc38a16eea	coupling, sign
HuggingFaceTB/SmolLM2-1.7B	effd688a1292	coupling, sign
allenai/OLMo-2-0425-1B	a1847dff3500	coupling, sign
allenai/OLMo-2-1124-7B	7df9a82518af	coupling, sign, checkpoint matrix
microsoft/Phi-4-mini-instruct	cfbefacb9925	coupling, sign
mistralai/Mistral-7B-v0.3	caa1feb0e54d	coupling, sign
ornith-ai/Ornith-1.5-9B	489cb97981b8	side information
nex-agi/Nex-N2-mini	ca218dcb1fbe	side information
GestaltLabs/Qwen3.6-35B-A3B-NSC-ACE-SABER-GGUF	eb964b1b7403	side information
DFloat11/Qwen3-8B-DF11	24545db8fabc	DFloat11 file sizes (headers)
DFloat11/Qwen3-14B-DF11	1b98c7539a10	DFloat11 file sizes (headers)

Table 10: Models used. The OLMo-2-7B checkpoint revisions (stage-1 steps, stage 2, instruct, OLMo-7B-0724) are in the data release, with full 40-character revisions.