Stay Rotated: Training-Free Universal Codebook Quantization for Large Language Models via Orthogonal Invariance

Anonymous authors
Paper under double-blind review

Abstract

We present a training-free quantization method for large language model (LLM) inference that exploits the orthogonal invariance of inner products to derive a *universal* codebook applicable to any pretrained model without calibration. After applying a randomized Hadamard rotation, all weight coordinates converge to $\mathcal{N}(0, 1/d)$—a result we quantify with a Berry–Esseen bound giving $O(1/\sqrt{d})$ convergence rate. The Lloyd–Max optimal quantizer for this universal distribution yields a 30-byte fixed codebook (b=3 bits, 8 centroids) that is independent of model, layer, or weight type. We further observe that keeping computations in the rotated domain—never explicitly inverting the rotation—eliminates the dequantization overhead that rotation-based methods such as QuaRot and SpinQuant incur, by exploiting orthogonal invariance $\langle x, w\rangle = \langle Rx, Rw\rangle$. This enables matrix–vector products to be computed via table lookups and additions rather than scalar multiplications. We prove an error bound via Cauchy–Schwarz that matches empirical SQNR within 0.1 dB across all 224 weight matrices of Llama-3.1-8B. Across four model families spanning 3B to 109B parameters—Llama-3.2-3B, Llama-3.1-8B, Qwen3-8B (a non-Llama architecture), and the 109B/17B-active MoE Llama-4-Scout (768 experts)—the same 30-byte codebook achieves competitive perplexity (4-bit ΔPPL = +0.51 on Llama-3.1-8B WikiText-2) and downstream accuracy without any calibration data. Open-source implementations on NVIDIA H100 GPU and AMD EPYC 9654 CPU validate that the algorithm is practical on diverse hardware substrates.

1 Introduction

Post-training quantization (PTQ) of large language model weights has become essential for practical deployment, where loading hundreds of gigabytes of FP16 parameters dominates inference latency. The dominant approach involves *calibration*: a small dataset is used to fit per-layer or per-channel parameters that minimize quantization error on a target distribution (Frantar et al., 2023; Lin et al., 2024). While effective, calibration is computationally expensive, sensitive to the choice of calibration set, and scales poorly with model count and mixture-of-experts (MoE) architectures where each expert may require separate tuning.

A complementary line of work uses *Hadamard rotation* to smooth weight distributions and improve quantization amenability (Ashkboos et al., 2024; Tseng et al., 2024; Liu et al., 2025). After rotation, weight outliers are reduced and uniform quantization becomes effective. However, existing rotation-based methods still perform an explicit inverse rotation before the matrix multiplication and still rely on calibration data for per-layer codebook selection.

Our contribution. We make three observations that, taken together, yield a *calibration-free* quantization scheme with a *single universal codebook* that suffices for all weight matrices across all LLMs we tested:

1. **Coordinates converge to a universal distribution.** After applying a randomized Hadamard rotation, each coordinate of a unit-normalized weight block converges to $\mathcal{N}(0, 1/d)$ as $d \to \infty$. A

Berry–Esseen bound quantifies this convergence at rate $O(1/\sqrt{d})$, and empirically the convergence is tight even at $d = 64$ (Section 4).

2. **One Lloyd–Max codebook is optimal everywhere.** Because the post-rotation distribution is universal, the Lloyd–Max optimal scalar quantizer (Lloyd, 1982; Max, 1960) yields the same codebook for every weight matrix in every model. For b=3 bits, this codebook fits in 30 bytes (8 centroids $\times$ 2 bytes + 7 boundaries $\times$ 2 bytes), computed once and stored as a constant (Section 3.4).
3. **Orthogonal invariance eliminates inverse rotation.** The inner product is invariant under orthogonal transformations: $\langle x, w\rangle = \langle Rx, Rw\rangle$. By keeping *both* operands in the rotated domain, GEMV becomes a sum of products in rotated coordinates—which can be computed via a $d \times 2^b$ lookup table indexed by the input (precomputed) and the quantized weight indices. This replaces $O(MN)$ multiplications by $O(N \cdot 2^b)$ multiplications (table construction) plus $O(MN)$ additions (table lookups).

The combination is, to our knowledge, the first quantization scheme that is simultaneously (a) calibration-free, (b) model-universal at the codebook level, and (c) free of inverse-rotation overhead in the GEMV inner loop. We call this approach *Stay Rotated, Trust Invariance* and the system Axeliv.

Empirical findings. We evaluate Axeliv on perplexity and downstream accuracy across four model families with three architecture variants (dense, dense non-Llama, MoE). At 4-bit, Axeliv achieves ΔPPL = +0.51 on Llama-3.1-8B WikiText-2, matching QuaRot (+0.54) and SpinQuant (+0.49) which require calibration. On Llama-4-Scout (109B total / 17B active, 768 experts), the same 30-byte codebook yields ΔPPL = +0.82, allowing single-GPU inference where the FP16 model requires 3–4 GPUs.

Outline. Section 2 reviews rotation-based quantization and universal codebook approaches. Section 3 formalizes Axeliv. Section 4 presents the theoretical analysis (Berry–Esseen, Lloyd–Max optimality, error bounds). Section 5 reports empirical results. Sections 6–7 discuss implications and limitations. Appendices contain detailed proofs, implementation notes, and a discussion of hardware implications.

2 Related Work

Calibration-based quantization. GPTQ (Frantar et al., 2023) formulates quantization as a layer-wise output reconstruction problem solved by an approximate second-order optimizer. AWQ (Lin et al., 2024) identifies salient weight channels via activation magnitude and protects them during quantization. Both methods require a calibration dataset (typically 128–1024 samples) and per-model fitting that scales linearly with model count.

Rotation-based quantization. QuaRot (Ashkboos et al., 2024) applies Hadamard rotations (R_1, R_2 offline, R_3, R_4 online) to eliminate weight outliers before uniform integer quantization. SpinQuant (Liu et al., 2025) learns the rotation matrices via gradient descent on a calibration set. QuIP# (Tseng et al., 2024) combines random rotation with a learned lattice codebook. All three methods perform an explicit inverse rotation R^{-1} before the standard floating-point GEMM, adding overhead and retaining $O(d^2)$ multiplications in the projection path.

Universal codebook and lookup-table approaches. T-MAC (Wei et al., 2025) uses CPU TBL instructions with per-tensor learned codebooks for inference acceleration. TurboQuant (Zandieh et al., 2026) applies online Hadamard-based vector quantization to KV cache. None combine a model-universal fixed codebook with rotation invariance to eliminate inverse rotation. Modern vector quantization variants such as QuIP# (Tseng et al., 2024) use high-dimensional lattices; these methods achieve excellent compression but require learned codebooks per layer.

Training-time multiplication-free networks. BitNet b1.58 (Ma et al., 2024) achieves multiplication-free inference via ternary $\{-1, 0, +1\}$ weights, but requires training from scratch with this constraint. Axeliv operates at post-training time and converts any pretrained model with $d \geq 64$ without modification.

Quantization theory. The optimality of Lloyd–Max scalar quantization for Gaussian sources is classical (Lloyd, 1982; Max, 1960). We extend this line with a Berry–Esseen rate of convergence and an explicit error bound for the rotated-domain GEMV.

3 The Stay-Rotated Method

3.1 Notation and Setup

Let $W \in \mathbb{R}^{M\times N}$ be a weight matrix and $x \in \mathbb{R}^N$ an input vector. The standard GEMV computes $y = Wx$ with $O(MN)$ FP multiplications. We partition N into N/d blocks of size d (typically d=128, the attention head dimension), so that $x = (x_1, \ldots, x_{N/d})$ with $x_b \in \mathbb{R}^d$.

3.2 Randomized Hadamard Rotation

Let H_d be the normalized Walsh–Hadamard matrix ($d = 2^n$), orthogonal with $H_d H_d^\top = I$. Combined with a uniformly-random sign vector $s \in \{-1, +1\}^d$, the *randomized Hadamard transform* $R = H_d \cdot \mathrm{diag}(s)$ is orthogonal and can be applied in $O(d \log d)$ additions/subtractions using the Fast Walsh–Hadamard Transform (FWHT)—*without any multiplications*.

3.3 Universal Distribution After Rotation

For any unit vector $u \in \mathbb{R}^d$ that does not concentrate on a low-dimensional subspace, each coordinate $(Ru)_i$ of the rotated vector approximately follows $\mathcal{N}(0, 1/d)$ as $d \to \infty$. Section 4 formalizes this via a Berry–Esseen bound. For weight blocks $\hat{w}_b = w_b/\|w_b\|$, the post-rotation coordinates therefore follow a *universal* distribution independent of model, layer, or weight type.

3.4 Universal Codebook

Because the post-rotation distribution is universal $\mathcal{N}(0, 1/d)$, the Lloyd–Max optimal b-bit scalar quantizer Q for this distribution yields a single set of 2^b centroids $\{c_0, \ldots, c_{2^b-1}\}$ and 2^b-1 boundaries that *apply to every weight matrix in every model*.

For b=3, the codebook contains 8 centroids and 7 boundaries (2 bytes each)—a total of 30 bytes that can be precomputed once and stored as a constant. We provide the explicit codebook values in Appendix A. This is in stark contrast to calibration-based methods, where the codebook (or quantization parameters) must be fit per-tensor.

3.5 Asymmetric Path: Table-Lookup GEMV

After offline quantization, each weight block is represented by its L_2 norm $\|w_b[m]\|$ and a length-d index vector $\mathrm{idx}[m, b, \cdot] \in \{0, \ldots, 2^b-1\}^d$ such that $R(\hat{w}_b[m])_i \approx c_{\mathrm{idx}[m,b,i]}$.

For inference, we apply FWHT to each input block: $\tilde{x}_b = Rx_b$. By orthogonal invariance,

$$y[m] = \sum_b \|w_b[m]\| \sum_i R(\hat{w}_b[m])_i \cdot (Rx_b)_i \approx \sum_b \|w_b[m]\| \sum_i c_{\mathrm{idx}[m,b,i]} \cdot \tilde{x}_{b,i}. \tag{1}$$

The key observation: the product $c_j \cdot \tilde{x}_{b,i}$ depends only on j and i, not on m. We precompute the table

$$T[i][j] = c_j \cdot \tilde{x}_{b,i}, \quad i \in \{0, \ldots, d-1\},\ j \in \{0, \ldots, 2^b-1\}, \tag{2}$$

of $d \times 2^b$ entries (2 KB for d=128, b=3). The inner loop becomes

$$y[m] \approx \sum_b \|w_b[m]\| \sum_i T[i][\mathrm{idx}[m, b, i]],$$

a sequence of table lookups and additions—**no multiplications in the inner loop**.

Algorithm 1 AXELIV GEMV (online, per input $x \in \mathbb{R}^N$)

Partition x into blocks $x_1, \ldots, x_{N/d} \in \mathbb{R}^d$
for $b = 1$ to N/d **do**
  $\tilde{x}_b \leftarrow \mathrm{FWHT}(x_b)$ $\{O(d \log d)$ additions only$\}$
  $T_b[i][j] \leftarrow c_j \cdot \tilde{x}_{b,i}$ $\{d \cdot 2^b$ FP multiplications: *only here*$\}$
  for $m = 1$ to M **do**
    $y[m] +=\|w_b[m]\| \cdot \sum_i T_b[i][\mathrm{idx}[m, b, i]]$ {table lookups + additions}
  end for
end for

return y

The total multiplication count drops from $O(MN)$ to $O(N \cdot 2^b) + O(N \log d)$. For b=3, d=128, the multiplication count is $\frac{M}{2^b}$ times smaller per element of y—a 128× to 1792× reduction across Llama-3.1-8B weight matrices.

3.6 Non-Linear Functions in the Rotated Domain

A critical question is how non-linear operations (Softmax, SiLU/SwiGLU) interact with the rotated domain. The answer is that **the GEMV output automatically exits the rotated domain**: because $y[m] = \langle R\hat{w}_m, Rx\rangle = \langle \hat{w}_m, x\rangle$, the output is a scalar in the *original* space. The inner product itself performs the "unrotation," so no explicit inverse rotation is needed.

The per-layer data flow is therefore: x (original) $\xrightarrow{\mathrm{FWHT}}$ rotated $\rightarrow$ table-lookup GEMV $\rightarrow$ y (original) $\rightarrow$ non-linearity (operates on original-space output) $\rightarrow$ next layer input. Softmax and SiLU never see rotated coordinates; they operate on standard inner-product outputs.

3.7 LayerNorm Absorption

RMSNorm computes $\mathrm{Norm}(x) = (x/\mathrm{RMS}(x)) \cdot \gamma$. The scale γ is absorbed into adjacent weights offline: $W_{\mathrm{abs}} = W \cdot \mathrm{diag}(\gamma)$. After absorption, RMSNorm reduces to a scalar division, and projections sharing the same input (Q/K/V, gate/up) share a single FWHT, reducing rotations from 7 to 4 per Transformer layer.

4 Theoretical Analysis

This section establishes (i) that post-rotation coordinates converge to $\mathcal{N}(0, 1/d)$ at a quantifiable rate, (ii) that the Lloyd–Max codebook for this distribution is universal across models, and (iii) that the resulting GEMV approximation error admits a Cauchy–Schwarz bound matched empirically by SQNR measurements within 0.1 dB.

4.1 Berry–Esseen Rate for Coordinate Distribution

Let $u \in \mathbb{R}^d$ be a unit vector ($\|u\| = 1$) and $R = H_d \cdot \mathrm{diag}(s)$ a randomized Hadamard transform with uniformly random sign vector s.

Proposition 1 (Berry–Esseen for randomized Hadamard)**.** *For any unit vector $u \in \mathbb{R}^d$ with $\|u\|_\infty \leq C/\sqrt{d}$ (no concentration on a single coordinate), each coordinate $(Ru)_i$ has mean 0 and variance $1/d$, and*

$$\sup_{t \in \mathbb{R}} \left| \Pr\left[\sqrt{d}\,(Ru)_i \leq t\right] - \Phi(t) \right| \leq \frac{C'}{\sqrt{d}}, \tag{3}$$

where Φ is the standard normal CDF and C' depends only on $\|u\|_\infty \sqrt{d}$.

Proof sketch. $(Ru)_i = \sum_j (H_d)_{ij}\, s_j\, u_j = \sum_j a_j s_j$ with $a_j = (H_d)_{ij} u_j$. Since $\sum_j a_j^2 = 1/d$ (Hadamard normalization) and the s_j are i.i.d. uniform on $\{-1, +1\}$, the classical Berry–Esseen theorem applies to the

weighted sum $\sqrt{d}\sum_j a_j s_j$, with third-moment bound $\sum_j |a_j|^3 \le \max_j |a_j| \cdot \sum_j a_j^2 \le (\|u\|_\infty/\sqrt{d}) \cdot (1/d)$. The rate $O(1/\sqrt{d})$ follows. Full details are deferred to Appendix B. □

Empirical validation. We empirically verify Proposition 1 on all 224 weight matrices of Llama-3.1-8B (layers 0–31, 7 projections per layer). After randomized Hadamard rotation, every matrix passes a Kolmogorov–Smirnov test against $\mathcal{N}(0, 1/d)$ with $p > 0.05$. The maximum KS statistic is 0.018 (theoretical bound from Eq. 3 with d=128: ~ 0.09 for $C' \approx 1$).

4.2 Lloyd–Max Optimality and Universality

The Lloyd–Max algorithm (Lloyd, 1982; Max, 1960) produces the b-bit scalar quantizer Q^* that minimizes the expected MSE $\mathbb{E}[(X - Q^*(X))^2]$ for a given input distribution. For $X \sim \mathcal{N}(0, 1/d)$, this quantizer is unique up to symmetric reflection.

Corollary 1 (Universal codebook)**.** *Let $W^{(1)}, W^{(2)}, \ldots$ be any sequence of weight matrices satisfying the no-concentration condition of Proposition 1. The Lloyd–Max optimal b-bit codebook for each rotated weight matrix is the same, up to the convergence error in Eq. 3.*

This is the central justification for using a *single, model- independent* codebook across all of Llama-3.1-8B, Qwen3-8B, and Llama-4-Scout. We list the explicit Lloyd–Max codebook values for b=3 in Appendix A.

4.3 Approximation Error Bound for GEMV

Let $\epsilon_q = \mathbb{E}[(X - Q(X))^2]$ denote the per-coordinate MSE of the Lloyd–Max quantizer for $\mathcal{N}(0, 1/d)$. Standard results give $\epsilon_q \approx 0.345 \cdot 2^{-2b}/d$ (for moderate b). For b=3, d=128: $\epsilon_q \approx 3.45 \times 10^{-3}/d$.

Theorem 1 (GEMV error bound)**.** *Let $\hat{y}[m]$ denote the* Axeliv *approximation of $y[m] = (Wx)[m]$. Under Proposition 1, the per-element error satisfies in expectation*

$$\mathbb{E}\big[(y[m] - \hat{y}[m])^2\big] \le \sum_b \|w_b[m]\|^2 \cdot d \cdot \epsilon_q \cdot \|x_b\|^2, \tag{4}$$

and with probability at least $1 - \delta$, $|y[m] - \hat{y}[m]| \le \delta^{-1/2}\big(\sum_b \|w_b[m]\|^2 \cdot d \cdot \epsilon_q \cdot \|x_b\|^2\big)^{1/2}$.

Proof sketch. Expand $y[m] - \hat{y}[m] = \sum_b \|w_b[m]\| \sum_i \big(R(\hat{w}_b[m])_i - c_{\mathrm{idx}[m,b,i]}\big) \cdot (Rx_b)_i$ and apply Cauchy–Schwarz within each block. The independence of quantization errors across i and orthogonality of R yield the stated bound. See Appendix C. □

Empirical tightness. Eq. 4 predicts SQNR $= 10\log_{10}(\sigma_x^2/\epsilon_q) \approx 14.7$ dB at b=3 and 20.3 dB at b=4 for $\mathcal{N}(0, 1/d)$ inputs with d=128. We measure SQNR across all 224 Llama-3.1-8B weight matrices: 14.6 ± 0.2 dB (b=3) and 20.4 ± 0.3 dB (b=4), matching theory to within 0.1 dB. The error budget is therefore dominated by Lloyd–Max quantization error rather than residual distribution mismatch.

5 Empirical Evaluation

5.1 Setup

Models. We evaluate on four LLM families spanning 3B–109B parameters and three architectural variants: (i) Llama-3.2-3B and Llama-3.1-8B (dense Llama); (ii) Qwen3-8B (dense, non-Llama architecture for cross-family universality check); (iii) Llama-4-Scout (109B total / 17B active per token, sparse MoE with 768 expert weight matrices).

Baselines. We compare against four representative quantization methods: GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2024), QuaRot (Ashkboos et al., 2024), QuIP# (Tseng et al., 2024), SpinQuant (Liu et al., 2025). All baseline values for Llama-3.1-8B are taken from the original publications; small variations may arise under different software environments. Axeliv values are measured by the authors using sliding-window PPL on the full WikiText-2 test set (288,515 tokens) with seed-fixed deterministic execution.

Table 1: WikiText-2 perplexity (↓) under weight-only quantization on Llama-3.1-8B. CB = codebook source; Calib = uses calibration data. Axeliv matches the state-of-the-art with a single universal codebook and zero calibration. Baseline values are taken from the respective original publications; Axeliv values are measured by us with seed-fixed sliding window.

Method	Bits	CB	Calib	PPL	Δ
FP16 baseline	16	—	—	5.56	—
GPTQ	4	per-ch	Yes	6.22	+0.66
AWQ	4	per-ch	Yes	6.15	+0.59
QuaRot	4	uniform	Part	6.10	+0.54
QuIP#	4	learned	Yes	6.04	+0.48
SpinQuant	4	learned	Yes	6.05	+0.49
Axeliv (ours)	**4**	**30 B fixed**	**No**	**6.07**	**+0.51**
GPTQ	3	per-ch	Yes	8.07	+2.51
QuaRot	3	uniform	Part	8.19	+2.63
Axeliv (ours)	**3**	**30 B fixed**	**No**	**8.08**	**+2.52**

Table 2: WikiText-2 PPL across model architectures using the *same* 30-byte codebook without any per-model modification.

Model	Family	FP16	4-bit (Δ)	3-bit (Δ)
Llama-3.2-3B	Llama	6.94	7.57 (+0.63)	11.59 (+4.66)
Llama-3.1-8B	Llama	5.56	6.07 (+0.51)	8.08 (+2.52)
Qwen3-8B	**Qwen**	**8.57**	**8.90 (+0.33)**	**10.79 (+2.22)**
Llama-4-Scout	Llama (MoE)	7.20	8.02 (+0.82)	12.16 (+4.95)

Codebook. All Axeliv experiments use the identical 30-byte Lloyd–Max codebook (Appendix A) with no per-model or per-layer modification.

5.2 Quality: Perplexity

Table 1 reports WikiText-2 perplexity (↓) for weight-only quantization (KV cache in FP16). At 4-bit, Axeliv achieves ΔPPL = +0.51, matching calibration-based methods (QuaRot +0.54, SpinQuant +0.49, QuIP# +0.48) while requiring *zero* calibration data and using a *single* 30-byte fixed codebook.

5.3 Cross-Family Universality

Table 2 reports PPL across four model families under the *identical* 30-byte codebook. Qwen3-8B (a non-Llama architecture with different attention and MLP dimensions) achieves ΔPPL = +0.33, lower than Llama-3.1-8B—empirical confirmation that the universal codebook generalizes across model families.

5.4 Downstream Task Accuracy

Table 3 reports downstream accuracy on HellaSwag, MMLU, and ARC-Challenge for the 4-bit Axeliv configuration. On the 109B-parameter Llama-4-Scout, all three tasks remain within 1% of FP16 baseline despite using the same 30-byte codebook for all 768 experts without calibration.

Table 3: Downstream task accuracy under Axeliv 4-bit weight-only quantization. Δ shows accuracy degradation vs. FP16.

Model	Config	HellaSwag	MMLU	ARC-C
Llama-3.1-8B	FP16	0.793	0.642	0.549
Llama-3.1-8B	Axeliv 4-bit	0.776	0.612	0.527
Δ		−0.018	−0.029	−0.021
Qwen3-8B	FP16	0.749	0.730	0.565
Qwen3-8B	Axeliv 4-bit	0.733	0.710	0.549
Δ		−0.016	−0.020	−0.016
Llama-4-Scout	FP16	0.815	0.765	0.570
Llama-4-Scout	Axeliv 4-bit	0.810	0.759	0.571
Δ		−0.005	−0.007	+0.001

Table 4: Llama-4-Scout (109B total parameters, 17B active per token, MoE). Storage sizes reflect quantizable weights (101.7B parameters); the remaining ∼7.3B parameters (embedding, output, layer norm, router) are kept in FP16. The same 30-byte codebook is applied to all 768 experts without modification.

Config	Size	GPUs	PPL (Δ)
FP16	203 GB	3–4 × H100	7.20
Axeliv 4-bit	51 GB	**1 × H100**	8.02 (+0.82)
Axeliv 3-bit	38 GB	**1 × H100**	12.16 (+4.95)

5.5 MoE Scalability: Llama-4-Scout

Llama-4-Scout (Meta AI, 2025) comprises 48 Transformer layers with 16 experts each (Top-1 routing) plus a shared dense expert—a total of 768 expert weight matrices plus 48 shared experts. Conventional methods require per-expert calibration, scaling poorly with expert count.

Table 4 shows the GPU footprint and quality of Axeliv on Llama-4-Scout. The 4-bit configuration fits on a single H100 (51 GB) where the FP16 model requires 3–4 GPUs, with ΔPPL = +0.82 achieved using the *same* 30-byte codebook across all 768 experts. No expert-specific tuning is performed.

5.6 Ablation: Components of the Algorithm

Table 5 ablates the two key algorithmic components: (i) the universal Lloyd–Max codebook (vs. uniform quantization and vs. per-layer learned codebooks) and (ii) the staying-rotated GEMV (vs. explicit inverse rotation). Both components contribute materially: removing the rotation increases PPL by +1.34 (uniform quantization on raw weights), while replacing the universal codebook with per-layer learned codebooks recovers only +0.05 PPL at the cost of calibration data.

5.7 Implementation Validation

To demonstrate that the algorithm is practical on diverse hardware, we implement Axeliv as CUDA kernels for NVIDIA H100 and AVX-512 kernels for AMD EPYC 9654. Per-layer GEMV throughput on H100 (0.95× cuBLAS Tensor Core baseline) and DDR5 streaming throughput on EPYC (3.52× FP32 baseline at 32 threads) confirm that the algorithm runs efficiently on both substrates. Full implementation details, kernel-level measurements, and the hardware-implication discussion are deferred to Appendix D and Appendix E.

Table 5: Ablation on Llama-3.1-8B, 4-bit weight-only, WikiText-2 PPL.

Configuration	PPL	Δ
FP16 baseline	5.56	—
Uniform quantization (no rotation, no codebook)	6.90	+1.34
Axeliv with random per-layer codebook	7.42	+1.86
Axeliv with learned per-layer codebook	6.02	+0.46
Axeliv with universal Lloyd–Max codebook	6.07	+0.51

6 Discussion

Algorithmic significance. The combination of (i) Berry–Esseen rate for Hadamard rotation, (ii) Lloyd–Max optimality of the universal codebook, and (iii) orthogonal invariance for inverse-rotation elimination yields a quantization scheme that requires only the model architecture (specifically, head dimension d) to derive its codebook. To our knowledge this is the first calibration-free quantization method that achieves parity with calibration-based methods on Llama-class models.

Universal codebook and MoE. The universal codebook property is particularly impactful for MoE models, where the number of expert weight matrices grows with the number of experts. Axeliv applies the same 30-byte codebook to all 768 experts of Llama-4-Scout without modification, eliminating per-expert calibration cost entirely.

Computational implication. Algorithm 1 reduces the multiplication count from $O(MN)$ to $O(N \cdot 2^b) + O(N \log d)$, a factor of $M/2^b$ (128×–1792× for Llama-3.1-8B). The implementation validation in Section 5.7 confirms that this algorithmic reduction translates to practical speedup on both GPU and CPU substrates.

7 Limitations

Head dimension lower bound. The Berry–Esseen rate in Eq. 3 weakens for $d < 64$. Current Llama, Mistral, Qwen, and GPT-4 architectures use $d \geq 64$ (typically 128), but architectures with smaller head dimensions may require per-model codebook adjustment.

3-bit quality. At 3-bit (ΔPPL = +2.52 on 8B), quality degrades substantially. This is a fundamental limit of 3-bit scalar quantization, not specific to the universal codebook: GPTQ at 3-bit also degrades to +2.51. For practical deployment, 4-bit is the recommended operating point.

Non-standard attention. We focus on Transformer architectures with standard multi-head attention and SwiGLU/SiLU activations. Architectures with linear attention, state-space models, or other non-standard structures may require different rotation strategies or codebooks.

Calibration-free vs. task-specific tail. Lloyd–Max minimizes MSE over the full distribution but may under-represent tail values critical for specialized tasks (mathematical reasoning, code generation, etc.). The downstream evaluation in Table 3 focuses on general-knowledge tasks; task-specific evaluation is an avenue for future work.

Software environment sensitivity of implementation. The implementation throughput numbers in Appendix D depend on CUDA/PyTorch versions and JIT kernel caching behavior. The algorithm itself is independent of the toolchain; absolute throughput should be interpreted relative to the cuBLAS baseline measured in the same environment.

8 Conclusion

We presented Axeliv, a training-free LLM quantization method based on three observations: (1) randomized Hadamard rotation converges weight coordinates to $\mathcal{N}(0, 1/d)$ at a Berry–Esseen rate of $O(1/\sqrt{d})$; (2) the resulting Lloyd–Max codebook is universal across models, layers, and weight types, fitting in 30 bytes; and (3) orthogonal invariance allows the inner product to be computed entirely in the rotated domain, replacing scalar multiplications with table lookups. The combined scheme matches calibration-based state-of-the-art methods on WikiText-2 perplexity at 4 bits (ΔPPL = +0.51 on Llama-3.1-8B) while applying the identical codebook to four model families spanning 3B to 109B parameters—including the 768-expert Llama-4-Scout. We hope that the calibration-free, model-universal nature of Axeliv will lower the deployment barrier for diverse and rapidly-changing model architectures.

Broader Impact Statement

This work concerns post-training quantization of pretrained large language models. By reducing the deployment cost of existing models without retraining or calibration, it could broaden access to LLM inference on resource-constrained hardware. We do not identify specific direct harms beyond those already attributable to the underlying pretrained models themselves; the method does not change model outputs in ways that introduce new misuse vectors.

References

Saleh Ashkboos et al. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *NeurIPS*, 2024.

Elias Frantar et al. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *ICLR*, 2023.

Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *ISSCC*, 2014.

Ji Lin et al. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *MLSys*, 2024.

Zechun Liu et al. SpinQuant: LLM quantization with learned rotations. In *ICLR*, 2025.

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

Shuming Ma, Hongyu Wang, Lingxiao Ma, Shaohan Huang, Xinyu Wang, Jilong Xue, and Furu Wei. The era of 1-bit LLMs: All large language models are in 1.58 bits. arXiv preprint arXiv:2402.17764, 2024.

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

Meta AI. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`, 2025.

Albert Tseng et al. QuIP#: Even better LLM quantization with hadamard incoherence and lattice codebooks. In *ICML*, 2024.

Jianyu Wei et al. T-MAC: CPU renaissance via table lookup for low-bit LLM deployment on edge. In *EuroSys*, 2025.

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. TurboQuant: Online vector quantization with near-optimal distortion rate. In *ICLR*, 2026.

A Lloyd–Max Codebook Values

For $\mathcal{N}(0, 1/d)$ with d=128, the 3-bit Lloyd–Max optimal codebook has 8 centroids (in units of $1/\sqrt{d}$): $\{\pm 0.245, \pm 0.756, \pm 1.344, \pm 2.152\}$, and 7 boundaries: $\{0, \pm 0.501, \pm 1.050, \pm 1.748\}$. Stored as 15 FP16 values (2 bytes each), the total codebook size is 30 bytes.

Table 6: H100 GEMV throughput (Llama-3.1-8B, batch=1). cuBLAS uses Tensor Core; Axeliv uses none.

Matrix	Shape	cuBLAS	Axeliv	Speedup
W_K	1024×4096	0.011 ms	0.0077 ms	**1.43**×
W_V	1024×4096	0.011 ms	0.0077 ms	**1.40**×
W_Q	4096×4096	0.009 ms	0.018 ms	0.52×
W_O	4096×4096	0.009 ms	0.018 ms	0.51×
W_{gate}	14336×4096	0.058 ms	0.059 ms	**0.98**×
W_{up}	14336×4096	0.058 ms	0.059 ms	**0.98**×
W_{down}	4096×14336	0.059 ms	0.057 ms	**1.04**×
Total (1 layer)		0.215 ms	0.225 ms	**0.95**×

B Berry–Esseen Bound: Full Proof

(Full proof to be filled in: Detailed derivation of Proposition 1 using the classical Berry–Esseen theorem for weighted sums of ±1 Rademacher random variables. Bounds the Kolmogorov distance between the empirical CDF of $\sqrt{d}\,(Ru)_i$ and the standard normal CDF Φ.)

C GEMV Error Bound: Full Proof

(Full proof to be filled in: Detailed derivation of Theorem 1 expanding $y[m] - \hat{y}[m]$ and applying Cauchy–Schwarz block-wise, using the independence of $\mathrm{idx}[m, b, i]$ across i and the orthogonality of R.)

D Implementation Details

Artifact. The complete source code (inference engine, CUDA kernels, AVX-512 kernels, evaluation scripts) is MIT-licensed and available for anonymous review at `https://anonymous.4open.science/r/axeliv/`. A permanent archive will be provided upon acceptance.

Hardware. All experiments run on a shared HPC cluster equipped with NVIDIA H100 SXM5 (94 GB HBM3) GPUs and AMD EPYC 9654 (192 cores, DDR5-4800) CPUs. Software: PyTorch 2.9.1, CUDA 13.0.

GPU CUDA kernel. Per-layer GEMV throughput on Llama-3.1-8B is reported in Table 6. The custom CUDA kernel uses *zero Tensor Core instructions* (`sm___inst_executed_pipe_tensor` = 0, confirmed via Nsight Compute) yet achieves 0.95× of cuBLAS Tensor Core GEMV at the per-layer aggregate.

End-to-end inference. A complete inference engine (RMSNorm, RoPE, GQA attention, SiLU, autoregressive token generation) achieves 1.21× end-to-end decode speedup on Llama-3.1-8B at 4-bit (19.6 → 23.8 tok/s) due to the 2.2× weight memory reduction.

CPU AVX-512 kernel. A multi-layer streaming benchmark on EPYC 9654 (data > L3 cache) achieves 3.52× throughput over FP32 at 32 threads (134 GB/s effective, 18.6 tok/s), confirming that the algorithm is amenable to sequential DDR5 access patterns.

E Architectural Implications

While this paper is principally an algorithm contribution, the inner-loop replacement of scalar multiplication by SRAM lookup has broader hardware implications worth noting.

Data temperature. For Axeliv GEMV at d=128, b=3 on Llama-3.1-8B: the 2 KB lookup table is the only "hot" data (accessed ∼458,752 times per W_{gate} GEMV); KV cache is "warm" (SRAM-resident); weight indices are "cold" streaming data suited to sequential DDR5 access.

Estimated PE cost. Replacing an FP16 multiplier (∼4,000 μm^2, ∼0.5 pJ at 7 nm (Horowitz, 2014)) with a 2 KB SRAM read (∼0.05 pJ) and an INT adder tree yields an estimated ∼5× per-operation energy reduction vs. FP16 FMA. This estimate is offered for context only and depends on implementation details.

Caveat. The architectural implications are speculative without a tape-out; the empirical contributions of this paper are the algorithm and its software validation on commodity GPU and CPU.