International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 04 | April – 2026 DOI:10.55041/ISJEM06670
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

A Review of Quantization Techniques for Large Language Models: From Post-Training Quantization to Extreme 1-Bit Methods

Smit Vaghasiya, Baisampayan Dey

Mentor: **Sinalben Patel**

Abstract

Quantization reduces the numerical precision of LLM weights and activations from floating-point to low-bit integer formats, making it possible to run large models on hardware that would otherwise be off-limits. This review covers post-training quantization (PTQ), quantization-aware training (QAT), mixed precision, extreme sub-1-bit methods, and Key-Value (KV) cache compression, drawing on transformer-based LLM research up to early 2026. Standard PTQ methods like GPTQ and SpQR achieve near-lossless compression at 3–4 bits, with memory reductions exceeding 4× on 175B-parameter models. BitNet b1.58 pushes further, matching full-precision performance at 1.58-bit ternary weights from 3B parameters onward. More recently, sub-1-bit PTQ frameworks like NanoQuant and LittleBit have made it possible to run 70B-scale models on consumer GPUs, while KV cache methods like TurboQuant and CommVQ support million-token contexts within reasonable memory limits. We also cover two underappreciated issues: how Chain-of-Thought reasoning models break under standard quantization, and how low-bit quantization can inadvertently reverse machine unlearning.

Index Terms—Large Language Models, Post-Training Quantization, Quantization-Aware Training, KV Cache, Sub-1-Bit Compression, Model Compression, Low-Bit Inference

I. INTRODUCTION

Large language models have grown substantially in size. Models with over 100 billion parameters are now common in research, and running them at full precision requires hundreds of gigabytes of memory—well outside what most researchers and organizations can afford. Getting these systems to run on more modest hardware, without degrading their performance, is one of the more pressing practical problems in modern NLP.

Quantization is the most direct answer to that problem. It replaces high-precision floating-point numbers (FP16 or FP32) with lower-bit integers (8-bit or below), shrinking model size, speeding up matrix operations, and cutting power consumption. The core techniques are post-training quantization (PTQ), which compresses an already-trained model using a small calibration dataset; quantization-aware training (QAT), which incorporates low-precision effects into the fine-tuning process; mixed precision, which applies different bit-widths to different parts of the model; and extreme low-bit methods targeting 1-bit or ternary representations.

More recently, the field has moved into KV cache compression—the KV cache has become the main memory bottleneck as context windows stretch to 128K and 1M tokens [1]. Reasoning models like DeepSeek-R1 have also introduced a new challenge: their long Chain-of-Thought

(CoT) pathways turn out to be unusually sensitive to quantization errors [3].

This review covers quantization techniques for LLMs from the foundational QAT and PTQ methods through to 2026 work on sub-1-bit matrix factorization, KV cache vector quantization, and reasoning model compression. Evidence is organized thematically to draw out patterns in efficiency, accuracy, and scalability.

II. METHODS

A. Search Strategy

Papers were identified through structured searches of Google Scholar, arXiv, IEEE Xplore, and the ACL Anthology using Boolean queries designed for reproducibility—for example: ("LLM" OR "Large Language Model" OR "Transformer") AND ("Quantization" OR "PTQ" OR "QAT" OR "KV Cache" OR "Sub-1-bit"). The search covered English-language publications from 2017 to 2026, spanning both pre-transformer foundations and current state-of-the-art

contributions. Additional papers were found through backward citation chaining from key survey articles.

B. Study Selection

45 papers were initially identified. After removing duplicates, 40 unique records were screened against eligibility criteria based on title and abstract. 10 papers were excluded for insufficient relevance to transformer-based LLM quantization or missing quantitative metrics. 30 papers were included in the final synthesis.

Eligibility criteria included: (1) LLM Focus—does the paper address quantization for large language models or transformer-based LLMs? (2) Quantization Method—does it describe a specific technique (PTQ, QAT, KV Cache, extreme low-bit)? (3) Empirical Results—does it report quantitative performance metrics (perplexity, accuracy, speed)? (4) Temporal Recency—was it published between 2017 and 2026? Several pre-2020 CNN-focused papers were retained as they provide foundational evidence on mixed precision and early QAT/PTQ that directly informs modern LLM work.

C. PRISMA-trAIce AI Utilization Disclosure

In keeping with 2026 PRISMA-trAIce guidelines, we disclose that large language models were used during manuscript preparation for literature assistance, summarization, and structural editing [4]. All AI-assisted content was reviewed by the authors, who manually verified claims, empirical metrics, and citations against the original source documents.

III. RESULTS

A. Characteristics of Included Studies

TABLE I provides a comparative snapshot of the 2023–2024 foundational methods and the 2025–2026 advances in sub-1-bit and KV cache quantization covered in this review.

TABLE I

Characteristics of Selected Key Studies

Study	Year	Type	Bits	Key Focus
Frantar et al.	2023	PTQ	3-bit	Second-order error minimization
Ma et al.	2024	Extreme	1.58-bit	Ternary weights
NanoQuant [11]	2026	PTQ (Extreme)	Sub-1-bit	Latent binary ADMM factorization
LittleBit [14]	2025	PTQ (Extreme)	0.1 BPW	Latent matrix factorization
TurboQuant [16]	2026	KV Cache	3-4 bits	Vector quantization for KV cache
CommVQ [17]	2025	KV Cache	1-2 bits	RoPE-commutative codebook
PM-KVQ [3]	2025	KV Cache	Mixed	Progressive block-wise for long-CoT
XQuant [19]	2025	KV Cache	Sub-1.4 bits	Cross-layer compression

B. Post-Training Quantization (PTQ) at 3–8 Bits

PTQ at 8 bits produces negligible perplexity degradation relative to full precision. At 3–4 bits, the losses are small but measurable—for example, 3-bit OPT-175B scores 8.68 perplexity on WikiText2 versus 8.34 at full precision, with zero-shot accuracy of 76.19% against 75.59%, and memory reduced by more than 4×. SmoothQuant achieves a 1.56× speedup and 2× memory reduction at 8 bits across 530B-parameter models. SpQR delivers 15–30% faster inference at 3.36–4.71 bits with under 1% perplexity degradation.

Foundational CNN results (4-bit PTQ yielding 75.23% top-1 accuracy on ImageNet for ResNet50, compared to 76.07% at full precision) are included for context but should be interpreted cautiously given the architectural differences from LLMs. Baseline approaches like round-to-nearest degrade quickly, but PTQ variants avoid this through outlier handling. Whether via per-channel scaling or sparse isolation, the ability to treat a small subset of weights differently is what separates near-lossless quantization from catastrophic failure.

C. Advanced Outlier Mitigation and Deviation Correction

Extreme outliers in feature channels are a core challenge. QuaRot addresses this with Hadamard-based orthogonal rotations that smooth irregular weight-activation distributions, making tensors quantizable without

precision loss [5]. DuQuant extends the idea by using a zigzag permutation strategy to balance outlier distribution across blocks, enabling stable 4-bit weight-activation quantization [5]. D2Quant targets a different problem—activation drift in sub-4-bit regimes—by introducing Deviation-Aware Correction (DAC), which applies mean-shift correction directly inside post-attention LayerNorm operations to prevent errors from compounding [8].

D. Sub-1-Bit and Extreme Low-Bit Quantization

BitNet b1.58 reduced weights to 1.58-bit ternary values, enabling integer-only operations and matching FP16 performance at 3B parameters and above. At 70B scale, this translates to 4.1× inference speedup and 71.4× energy savings.

By 2025–2026, research moved past even that. NanoQuant frames compression as a low-rank binary factorization problem solved with the Alternating Direction Method of Multipliers (ADMM), bypassing traditional scalar quantization entirely. The result: a 70B model compressed 25.8× (from 138 GB to 5.35 GB) using just 128 calibration samples, runnable on a consumer 8 GB GPU at 20 tokens per second [11]. LittleBit goes further, targeting 0.1 bits per weight through Dual Sign-Value-Independent Decomposition (Dual-SVID) and latent matrix factorization, achieving 31× memory reduction [13].

Theme	Key Finding	Population	Effect
Extreme Low-Bit	Matches FP16 from 3B parameters with large energy savings	LLMs (3B–70B)	Positive
Sub-1-Bit PTQ	Compresses 70B models by >25x for 8GB consumer GPUs [11]	LLMs (13B–70B)	Positive
KV Cache Compression	87.5% cache size reduction at 2-bit; 128K context on single GPU [17]	Long-context LLMs	Positive
Reasoning Model Quantization	Up to 8% improvement over standard methods by controlling cumulative error [3]	Long-CoT models	Positive
PTQ at 3-8 Bits	>4x memory reduction for 175B models with negligible perplexity loss	LLMs (7B–175B)	Positive

E. Key-Value (KV) Cache Compression

As context windows expanded to 128K and beyond, the KV cache became the dominant memory constraint at inference time [15]. TurboQuant compresses KV cache data to 3–4 bits per value using training-free, data-oblivious vector quantization, delivering 6× memory reduction and 8× speedup on attention computation with no measured accuracy loss [1].

Rotary Position Embeddings (RoPE) create a specific mathematical problem for vector quantization—CommVQ solves it by designing a commutative quantization approach with an Expectation-Maximization learned codebook [17]. At 2-bit precision, CommVQ shrinks FP16 KV cache size by 87.5%, enabling Llama 3.1 8B to handle a 128,000-token context on a single RTX 4090 [18]. For million-token contexts, ParisKV uses CPU offloading via Unified Virtual Addressing (UVA), cutting decode latency by up to 44× versus prior baselines [15]. XQuant pushes KV cache compression to sub-1.4 bits through cross-layer compression [19].

F. Quantizing Reasoning and Long-CoT Models

Reasoning models like DeepSeek-R1 and Qwen 3 break under standard quantization in ways that simpler models do not [3]. When a model generates thousands of Chain-of-Thought tokens, small quantization errors accumulate across the chain, and short-context calibration data fails to capture the distribution. The result is logical degradation that goes beyond simple accuracy drops [3].

PM-KVQ (Progressive Mixed-Precision KV Cache Quantization) addresses this by assigning higher bit-widths to mathematically sensitive transformer blocks and using positional interpolation for calibration [3]. Compared to standard baselines, PM-KVQ improves reasoning benchmark performance by up to 8% while maintaining throughput [3].

G. Summary of Evidence

TABLE II summarizes practical outcomes across the main quantization categories reviewed here.

TABLE II

Summary of Evidence and Practical Impacts

IV. DISCUSSION

A. Evaluation Shifts: RULER and Reasoning Benchmarks

WikiText2 perplexity has long been the default metric for quantization quality. It no longer tells the whole story. Long-context evaluations show that 8-bit quantization holds up well, but 4-bit methods can cause dramatic accuracy drops—up to 59%—on retrieval tasks using the RULER and ONERULER benchmarks [21]. The field has responded by moving toward harder tests: GPQA-Diamond and MATH-500 better capture whether a quantized model can still reason coherently under pressure [23].

B. Security Risks: Quantization-Induced Unlearning Reversal

One finding from this review deserves particular attention. When LLMs undergo machine unlearning to remove copyrighted or toxic training data, that data is typically masked rather than fully deleted [24]. Applying 4-bit PTQ introduces mathematical noise that disrupts those masking pathways. Empirically, unlearned models that retain only 21% of forbidden data in full precision jump back to 83% retention after quantization—effectively undoing the unlearning process [24]. This has obvious implications for any deployment pipeline where quantization follows unlearning.

C. Practical Implications

For developers working within GPU memory constraints, PTQ methods offer the clearest path to single-GPU inference for 100B+ models. Sub-1-bit and extreme KV cache quantization methods now extend this to 70B+ reasoning models on consumer hardware and edge devices [11]. The tradeoffs are real: specialized INT4/binary kernel support is not universal across hardware, and the unlearning reversal issue affects any model where low-precision compression follows privacy or safety scrubbing.

V. CONCLUSION

Quantization has made very large language models deployable in contexts that would have been impractical two years ago. Early PTQ methods achieved near-lossless 3–4 bit compression. The 2025–2026 wave added sub-1-bit matrix factorization (NanoQuant, LittleBit) and aggressive KV cache compression (TurboQuant, CommVQ) that supports million-token contexts on edge hardware. As reasoning models with long CoT chains become more common, specialized quantization approaches like PM-KVQ are needed to prevent logical degradation.

Three gaps remain worth flagging: hardware support for binary and sub-1-bit kernels is still inconsistent; most evaluations still underuse long-context benchmarks; and the security implications of quantizing models that have undergone machine unlearning are only beginning to be understood.

. REFERENCES

[1] A. Zandieh et al., "Online Vector Quantization with Near-optimal Distortion Rate (TurboQuant)," ICLR, 2026.

[2] Z. Liu et al., "LLM-QAT: Data-free quantization aware training for large language models," arXiv:2305.17888, 2023.

[3] T. Shi et al., "PM-KVQ: Progressive Mixed-precision KV Cache Quantization for Long-CoT LLMs," OpenReview, 2025.

[4] PRISMA-trAIce, "Reporting guidelines for AI-assisted systematic reviews," 2026.

[5] A. Ashkboos et al., "QuaRot: Hadamard-based Rotation for Outlier Mitigation," 2024.

[6] T. Dettmers et al., "LLM.int8(): 8-bit matrix multiplication for transformers at scale," NeurIPS, 2022.

[7] K. Wang et al., "HAQ: Hardware-aware automated quantization with mixed precision," CVPR, 2019.

[8] X. Yan et al., "D2Quant: Deviation-Aware Correction for Accurate Low-Bit PTQ," 2026.

[9] Z. Yao et al., "ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers," NeurIPS, 2022.

[10] T. Dettmers et al., "SpQR: A sparse-quantized representation for near-lossless LLM weight compression," ICLR, 2024.

[11] Wang et al., "NanoQuant: Efficient Sub-1-Bit Quantization of Large Language Models," 2026.

[12] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," CVPR, 2018.

[13] B. Lee et al., "LittleBit: Ultra Low-Bit Quantization via Latent Factorization," NeurIPS, 2025.

[14] B. Lee et al., "LittleBit: Ultra Low-Bit Quantization via Latent Factorization," NeurIPS, 2025.

[15] A. Mekala et al., "Does quantization affect models' performance on long-context tasks?" EMNLP, 2025.

[16] A. Zandieh et al., "Online Vector Quantization with Near-optimal Distortion Rate (TurboQuant)," ICLR, 2026.

[17] J. Li et al., "CommVQ: Commutative Vector Quantization for KV Cache Compression," ICML, 2025.

[18] J. Li et al., "CommVQ: Commutative Vector Quantization for KV Cache Compression," ICML, 2025.

[19] H. Yang et al., "XQuant: Achieving Ultra-Low Bit KV Cache Quantization with Cross-Layer Compression," EMNLP, 2025.

[20] M. Nagel et al., "Up or down? Adaptive rounding for post-training quantization," ICML, 2020.

[21] A. Mekala et al., "Does quantization affect models' performance on long-context tasks?" EMNLP, 2025.

[22] J. Lin et al., "DuQuant: Outlier-aware Rotation Matrices and Channel Permutation," 2024.

[23] E. Frantar et al., "GPTQ: Accurate post-training quantization for generative pre-trained transformers," ICLR, 2023.

[24] Anonymous, "Quantization-induced Unlearning Reversal in LLMs," OpenReview, 2026.

[25] H. Wang et al., "BitNet: Scaling 1-bit transformers for large language models," arXiv:2310.11453, 2023.

[26] G. Xiao et al., "SmoothQuant: Accurate and efficient post-training quantization for large language models," ICML, 2023.

[27] A. Gholami et al., "A survey of quantization methods for efficient neural network inference," 2021.

[28] S. Ma et al., "The era of 1-bit LLMs: All large language models are in 1.58 bits," arXiv:2402.17764, 2024.