Azerbaijan Journal of High Performance Computing, Vol. 8, Art. e2026.01, 2026, pp. 1-19
https://doi.org/10.32010/26166127.2026.01

Effects of Quantization on Large Language Models' performance on Sentiment Analysis tasks

Aydın Gasimov
Azerbaijan State Oil and Industry University, Baku, Azerbaijan, aydin.gasimov@protonmail.com

*Correspondence: Mirakram Aghalarov, Baku Higher Oil School, Baku, Azerbaijan, aydin.gasimov@protonmail.com

Abstract

This paper investigates the effects of weight-only quantization on large language models (LLMs) for sentiment classification. We evaluate three representative models—Gemma 3 4B, Microsoft Phi-4 Mini, and Liquid LFM2 1.2B—across multiple quantization levels (FP16, Q8, Q6, Q5, Q4, Q4-QAT, Q3) using the IMDB, SST-2, and Twitter Airline Sentiment datasets. Our results show that sentiment classification is unusually resilient to quantization: accuracy differences relative to Q8 are typically within a few percentage points, with significant degradation only at 3-bit precision. Notably, Gemma's QAT-trained Q4 variant surpasses its higher-precision baselines, underscoring the promise of training-time adaptation. However, we also observe shifts in prediction behavior, including increased class polarization and diminished recognition of minority classes under lower precisions. From a systems perspective, quantization primarily yields storage and memory reductions on current GPUs lacking sub-8-bit execution, but delivers real runtime gains on processors with native INT4/INT2 or FP4 support, such as Qualcomm's Snapdragon 8 Elite Gen 5, NVIDIA Blackwell, and AMD/Intel NPUs. These findings highlight that while compact LLMs at 4-bit precision already offer an attractive efficiency–accuracy trade-off for on-device deployment, the full benefits of aggressive quantization will only be realized as native low-bit hardware becomes pervasive.

Keyword: Sentiment Analysis, Large Language Models, Quantization, Zero-shot learning, Compact LLMs, Edge AI

1. Introduction

Large language models (LLMs) have become central to natural language processing (NLP), enabling major advances in tasks such as sentiment analysis, translation, and summarization. Yet, their rapidly increasing scale introduces practical limitations (Vaswani et al., 2017). Models with billions of parameters require specialized hardware, consume significant memory, and impose high energy costs, creating barriers to deployment in settings where efficiency and accessibility are as important as raw predictive power (Kaplan et al., 2020). For sentiment analysis in particular—a

task widely applied in domains such as finance, healthcare, and customer engagement—the ability to run accurate models on commodity hardware is often more valuable than achieving marginal improvements with resource-intensive architectures.

Model compression has emerged as a promising strategy to address these limitations. Among different compression techniques, quantization—the conversion of model weights and activations to lower numerical precision—offers one of the most effective trade-offs. By reducing storage requirements and accelerating inference, quantization can enable models to run efficiently on CPUs, GPUs with limited VRAM, or even edge devices. However, the impact of quantization is not uniform. While some architectures tolerate aggressive reductions such as 4-bit quantization with minimal accuracy loss, others degrade sharply. The effectiveness of quantization also depends on the task: classification tasks like sentiment analysis may show different resilience compared to generative or reasoning tasks (Han et al., 2015).

Recent evaluations of compact models in the 3–4B parameter range have shown that Gemma 3 4B (Team et al., 2025) and Microsoft Phi-4 Mini (Abdin et al., 2024) consistently deliver strong performance on sentiment analysis benchmarks while maintaining substantially lower computational cost than their 7–8B counterparts. At the same time, smaller models such as Liquid LFM2 1.2B (Liquid AI, 2025) demonstrate potential for high-throughput applications, albeit with reduced accuracy on more nuanced datasets. These observations raise an important question: how do different levels of quantization alter the balance between accuracy, latency, and memory efficiency across models of different scales?

This paper addresses that question by systematically evaluating the effects of quantization

on three representative LLMs for sentiment analysis: Gemma 3 4B, as a strong mid-sized performer; Microsoft Phi-4 Mini, as another compact but effective baseline; and Liquid LFM2 1.2B, as a lightweight efficiency-oriented model. Each is tested under multiple quantization levels, including high-precision FP16, moderate compression with Q8, and aggressive compression with Q4 and Q4-QAT variants.

By narrowing the study to these models and datasets such as SST-2, IMDB, and Twitter Airline Sentiment, the goal is to provide a clear view of how quantization affects sentiment analysis performance in realistic deployment scenarios. Specifically, this work seeks to determine which quantization strategies preserve sufficient accuracy while enabling faster, more memory-efficient inference, and whether compact models remain the optimal choice under aggressive compression.

2. Related Work

Early work on transformer quantization in NLP concentrated on BERT-style encoders and already showed that substantial compression is possible with minimal accuracy loss on GLUE tasks, including the SST-2 sentiment benchmark. Q8BERT used quantizationaware training (QAT) to compress BERT to INT8 with approximately 4× smaller footprint and less than 1% average drop; its SST-2 accuracy remained virtually unchanged relative to FP32, establishing that sentiment classification can be robust to 8-bit quantization (Zafrir et al., 2019). Q-BERT extended this line by

combining group-wise quantization with Hessian-guided mixed precision, reaching as low as 2–3 bits and still keeping SST-2 within a few points of the baseline, while analyzing which modules are most sensitive to aggressive quantization (Shen et al., 2020).

As models shifted from encoders to large generative LLMs, post-training quantization (PTQ) became the dominant approach for rapid deployment. GPTQ introduced an efficient one-shot, second-order PTQ that reliably delivers 3–4-bit weight quantization with small quality drops across GPT/OPT-like models, enabling substantial memory and throughput gains (Frantar et al., 2022). SmoothQuant addressed activation outliers to enable hardware-friendly W8A8 quantization, improving latency and memory while preserving accuracy across many LLM families (Xiao et al., 2023). AWQ proposed activation-aware protection of salient weight channels for low-bit (e.g., INT3/4) weight-only PTQ that remains hardware-efficient; open-source implementations have made AWQ a common baseline in practice (Lin et al., 2024). Additional PTQ directions include SpQR, which mixes sparsity and quantization to approach near-lossless compression (Dettmers et al., 2023), and SqueezeLLM, which combines sensitivity-based non-uniform bit assignment with dense-and-sparse decomposition to reach ~3-bit regimes with modest perplexity impact (Kim et al., 2023). Recent systematizations study how to combine these techniques and extend them to alternative numerical formats or mixed-precision attention blocks (Zhu et al., 2024).

Empirical evaluations that span multiple families and tasks consistently find that sensitivity to quantization varies by task, with NLI typically more robust than knowledge-heavy QA; importantly for this paper, several studies report that sentiment classification (e.g., GLUE/SST-2) is among the more resilient tasks, sometimes even showing small gains at 4-bit due to regularization-like effects (Jin et al., 2024)(Li et al., 2024). For example, Li et al. evaluate PTQ across 11 LLM families and multiple dimensions (weights, activations, and KV cache) and observe modest drops at 3–4 bits, with some zero-shot settings on GLUE-SST improving at 4-bit; their benchmark and toolbox underscore the importance of calibration data and method choice (Jin et al., 2024).

Low-bit inference also interacts with memory bottlenecks beyond model weights. The KV cache, which grows with sequence length and batch size, has become a prime target: KIVI demonstrates tuning-free 2-bit KV cache quantization with little quality loss and large throughput gains (Zhang et al., 2024); KVQuant shows that 2-bit KV compression can unlock milliontoken contexts on commodity setups (Hooper et al., 2024); subsequent work continues to refine importance and quantile-aware schemes for stable long-context inference (Hooper et al., 2024).

Adjacent to inference-time PTQ, training-time and fine-tuning oriented approaches also contribute to the landscape. LLM.int8() introduced outlier-aware 8-bit matrix multiplication that retains FP16-level quality during inference and loading, halving memory for projection layers (Dettmers et al., 2022). QLoRA popularized low-rank adaptation on top of 4-bit-quantized base weights, enabling resource-frugal fine-tuning with task-level quality comparable to full-precision finetuning—relevant when building sentiment-specific adapters under tight memory budgets (Dettmers et al., 2023).

Surveys synthesizing these advances emphasize three trends pertinent to sentiment analysis: 1) method-task interactions matter—PTQ choice, calibration data, and bit allocation can change outcomes across datasets; 2) memory wins increasingly come from the KV cache as much as from weight quantization; and 3) practical toolchains (e.g., AWQ/AutoAWQ) have matured, lowering engineering overhead for controlled studies like ours (Zhu et al., 2024)(Hansen et al., 2023).

Positioning: Building on this literature, our study specifically targets sentiment classification with instruction-following LLMs under deterministic decoding, comparing several quantization regimes (e.g., Q8/Q6/Q4/Q3) and training settings on canonical datasets (SST-2, IMDB, Twitter Airline). Prior BERT-era findings and recent LLM-wide evaluations suggest that sentiment tasks are comparatively robust to quantization; our contribution is to validate (or qualify) this claim systematically across modern LLMs and quantizers, with attention to practical inference details—such as polarization—that materially affect accuracy and throughput in realistic pipelines.

3. Methodology

All experiments were carried out using the llama.cpp inference framework, which provides consistent support for multiple weight-only quantization formats through the GGUF file standard. A lightweight Python script was used to automate dataset loading, inference, and evaluation, ensuring consistency across all runs.

Inference was conducted in a zero-shot setting with instruction-tuned checkpoints. Each input was wrapped in a short prompt explicitly instructing the model to classify the sentiment (e.g., “Answer with one of: positive, negative”).

Decoding followed a deterministic greedy strategy: temperature was fixed at T=0, reducing the softmax distribution

$$P(y = i|x) = \frac{e^{z_i}}{\sum_{j=i}^{C} e^{z_j}} \quad (1)$$

to simple argmax decoding (Vaswani et al., 2017)

$$y_t = arg\ max_i\ z_{t,i} \quad (2)$$

thereby eliminating stochastic variation across runs.

In addition to accuracy, evaluation included Matthews Correlation Coefficient (MCC)
across all datasets, defined as

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (3)$$

where TP, TN, FP, FN denote true positives, true negatives, false positives, and false negatives (Matthews, 1975). MCC is particularly valuable as it provides a balanced measure even when class distributions differ.

For the Twitter Airline Sentiment Dataset, which exhibits significant class imbalance, we also reported the Macro-F1 score, computed as the unweighted average of F1 across classes:

$$MacroF1 = \frac{1}{C}\sum_{i=1}^{C} \frac{2 \cdot P_i \cdot R_i}{P_i + R_i} \quad (4)$$

with C denoting the number of classes, $P_i = \frac{TP_i}{TP_i + FP_i}$ the precision, and $R_i = \frac{TP_i}{TP_i + FN_i}$

the recall for class i, respectively.(Wang et al., 2018)

3.1. Datasets and Models used

3.1.1. Datasets

- IMDB Reviews Dataset: A dataset of movie reviews labeled by overall sentiment. In this study, a stratified sample of 10,000 reviews was used to ensure balanced class representation while keeping runtime feasible. The longer review texts make this dataset a strong test of model performance on extended inputs. (Maas et al., 2011)
- Stanford Sentiment Treebank (SST-2): A benchmark dataset for binary sentiment classification consisting of short movie review phrases labeled as positive or negative. It is widely used for evaluating sentence-level sentiment analysis. (Socher et al., 2013)(Wang et al., 2018)

Twitter Airline Sentiment Dataset: A collection of tweets directed at US airlines, labeled as positive, negative, or neutral. Its imbalanced class distribution makes it valuable for testing robustness under skewed data. (Crowdflower, 2015)

3.1.2. Models

- Gemma 3 4B: A mid-sized instruction-tuned LLM developed by Google, designed to balance accuracy and efficiency. It delivers strong results on classification tasks at moderate scale (Team et al., 2025).
- Phi-4 Mini: A compact model in Microsoft's Phi series, optimized for efficiency and low resource usage. Despite its smaller size, it achieves competitive performance on sentiment analysis benchmarks. (Abdin et al., 2024)
- Liquid LFM2 1.2B: A lightweight foundation model from Liquid AI, designed for on-device deployment and resource-constrained settings. Its reduced parameter count emphasizes throughput and memory efficiency, though with some trade-offs in accuracy on nuanced tasks (Liquid AI, 2025).

3.2. Quantization Schemes (llama.cpp / GGUF)

We study weights-only post-training quantization as implemented in llama.cpp (GGUF).

In our experiments, Q8 serves as the baseline for all models; F16 is available for one model and is included for reference. Llama.cpp uses blockwise linear quantization: weights are partitioned into fixed-size blocks and de/quantized via linear mappings with block-local scale (and, for some schemes, an offset). Activations remain floating-point in llama.cpp. (Han et al., 2015)

General formulation. For a block B with weights ${w_i}^B$, quantization uses

$$ {q_i}^B = clip(round(\frac{{w_i}^B - b_B}{a_B}), q_{min}, q_{max}), \quad \widehat{\ } {w_i}^B = a_B {q_i}^B + b_B \tag{5} $$

where ${w^B}_i$ denotes the i-th weight in block $B, {q^B}_i$ is the corresponding quantized integer, and ${\hat{w}_i}^B$ is the reconstructed (dequantized) weight. (Han et al., 2015) a_B is the block-wise scale, b_B is an optional offset (zero-point), and $[q_{min}, q_{max}]$ defines the integer range determined by the bit-width.

The function $clip(\cdot)$ constrains values to this range.

3.2.1. F16 (half precision, reference only for one model)

Weights are stored as IEEE FP16 (no integer quantization). This provides a higher precision reference relative to integer formats, but with larger memory than Q8/Q6/Q5/Q4/Q3. GGUF supports FP16 tensors alongside quantized ones.

3.2.2. Q8_0 (8-bit, blockwise, symmetric)

A classic 8-bit symmetric scheme in llama.cpp with block size 32: each block stores a floating-point scale plus 32 signed 8-bit codes. Reconstruction is given by

$$\widehat{w_i}^B = s_B \cdot {q_i}^B \tag{6}$$

where ${q_i}^B$ is the quantized integer value and s_B is the block-wise scale.

For symmetric quantization schemes such as Q8_0, the offset term is zero $(b_B = 0)$, reducing reconstruction to a simple scaling operation.

This preserves accuracy close to FP baselines while cutting memory substantially. We use Q8 as our baseline in all comparisons.

3.2.3. K-quants (super-block formats): Q6_K, Q5_K_M, Q4_K_M, Q3_K_L

The K family in llama.cpp groups weights into super-blocks (typically 256 elements) composed of sub-blocks, with one or two floating-point scales per super-block to improve local dynamic range at lower bit-widths. De/quantization still uses linear maps as above, but the scale/offset layout differs from simple 32-element blocks to reduce error at 6–3 bits.

Q6_K (6-bit K-quant). Near-baseline quality with markedly lower memory than Q8_0 by leveraging super-block scaling. $\widehat{w_i}^B = a_B {q_i}^B$ (or $a_B {q_i}^B + b_B$ where applicable)

Q5_K_M (5-bit K-quant, “M” mix). A 5-bit K variant using a mixed perlayer recipe (“S/M/L” denote how aggressively sensitive tensors are given higher-bit treatment). “M” is the middle recipe and commonly recommended.

Q4_K_M (4-bit K-quant, “M” mix). 4-bit K with the “M” recipe; empirically superior to older non-K 4-bit schemes at similar size thanks to the super-block scaling.

Q3_K_L (3-bit K-quant, “L” mix). Heaviest compression among our settings.

The “L” recipe assigns higher-bit K types to a few critical matrices (e.g., attention/FFN outputs) and Q3_K elsewhere, trading extra bytes for better quality than a uniform 3-bit assignment.

Note on “S/M/L” suffixes: these are preset mixes across layers/tensors, not different math. They determine which tensors receive higher-bit K types within a single file for a better quality/bit-per-weight trade-off.

3.2.4. Q4_0 (4-bit, blockwise, symmetric)

An earlier 4-bit symmetric scheme using 32-element blocks with a single scale (no persuper-block refinement). While very compact, it typically shows a larger quality drop

than K-quants at the same nominal bit-width:

$$\widehat{w_i}^{B} = a_B q_i{}^{B} \tag{7}$$

3.2.5. Quantization-Aware Training (QAT) for Q4_0 (Gemma only)

To mitigate the accuracy loss of naïve 4-bit post-training quantization, we also evaluate QAT-trained checkpoints (available for Gemma 3) whose GGUF weights are stored as Q4_0 but were trained with simulated quantization in the loop. During QAT, forward passes insert a fake-quantization operator:

$$fq(w) = a_B \cdot clip(round(\frac{w-b_B}{a_B}, q_{min}, q_{max}) + b_B, \tag{8}$$

and gradients use a straight-through estimator (STE) to propagate gradients through the non-differentiable quantization operator (Han et al., 2015).

$$\frac{\partial fq(w)}{\partial w} \approx 1 \tag{9}$$

allowing weights to adapt to quantization noise. At inference time, the stored Q4_0 weights dequantize linearly as usual. Empirically, QAT reduces the perplexity drop compared with naïve post-training Q4_0, approaching BF16 quality while using roughly a quarter of the memory. In our study, QAT (Q4_0) is therefore only present for Gemma.

Scope note: All schemes above quantize weights only; activations and token computations remain floating-point in llama.cpp, which simplifies deployment and avoids runtime numerical regressions unrelated to weight format.

4. Results

We organize results by dataset, metric, and model, reporting performance under multiple weight-only quantization levels. For each dataset, we additionally compute the accuracy change (in %) relative to the Q8 baseline. To visualize the trade-off between accuracy and memory footprint, we plot accuracy on the Y-axis against memory usage on the X-axis; the X-axis is inverted so that lower memory usage appears further to the right.

4.1. IMDB Dataset

IMDB accuracies for each quantization level are given in Table 1, while the corresponding

MCC values are reported in Table 2. Accuracy differences relative to the Q8 baseline are summarized in Table 3. Figure 1 visualizes the accuracy–memory relationship for this dataset.

The IMDB dataset demonstrates stable performance as precision is reduced from 8-bit to 4-bit, with a pronounced decline only at the Q3 quantization level (Tables 1–3 and Figure 1). Notably, the Gemma-3-4B QAT variant outperforms its Q8 baseline, reflecting the benefit of additional training under quantization constraints (Table 3).

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.896	0.901	—	0.901	0.935	0.857
Phi 4 Mini	—	0.925	0.928	—	0.924	—	0.914
LFM2 1.2B	0.923	0.921	0.922	0.923	0.922	0.918	—

Table 1: IMDB dataset accuracies

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.803	0.812	—	0.810	0.870	0.740
Phi 4 Mini	—	0.850	0.857	—	0.849	—	0.830
LFM2 1.2B	0.846	0.842	0.844	0.846	0.846	0.836	—

Table 2: IMDB dataset MCC scores

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.0	+0.56	—	+0.56	+4.35	-4.35
Phi 4 Mini	—	0.0	+0.32	—	- 0.11	—	-1.19
LFM2 1.2B	+0.22	0.0	+0.11	+0.22	+0.11	-0.33	—

Table 3: Accuracy difference (%) vs. Q8 baseline for IMDB. Positive values indicate improvement; negative values indicate degradation

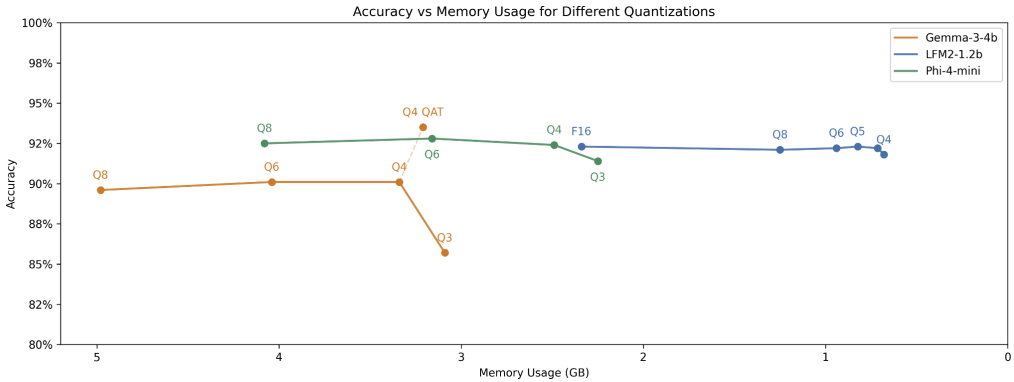

Fig. 1. Accuracy changes by quantization level for the IMDB dataset.

4.2. Stanford Sentiment Treebank (SST-2)

SST-2 accuracies are shown in Table 4, with MCC values in Table 5. Relative accuracy differences vs. Q8 is summarized in Table 6, and the accuracy–memory trend is shown in Figure 2.

The SST-2 dataset remains largely stable down to 4-bit quantization (Tables 4–6 and Figure 2), with a marked drop primarily at Q3 for Gemma. Small apparent improvements for Phi-4 Mini at Q3 and for LFM2 at $Q4_{k\_m}$ (Table 6) are likely attributable to stochastic variation rather than systematic gains from lower precision.

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.930	0.932	—	0.928	0.943	0.853
Phi 4 Mini	—	0.932	0.935	—	0.923	—	0.937
LFM2 1.2B	0.903	0.904	0.906	0.911	0.922	0.913	—

Table 4: SST-2 dataset accuracies

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.866	0.870	—	0.861	0.887	0.738
Phi 4 Mini	—	0.867	0.871	—	0.850	—	0.875
LFM2 1.2B	0.807	0.809	0.814	0.823	0.844	0.826	—

Table 5: SST-2 dataset MCC scores

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.0	+0.25	—	-0.25	+1.36	-8.26
Phi 4 Mini	—	0.0	+0.25	—	-0.98	—	+0.49
LFM2 1.2B	-0.13	0.0	+0.25	+0.76	+2.03	-1.02	—

Table 6: Accuracy difference (%) vs. Q8 baseline for SST-2. Positive values indicate improvement; negative values indicate degradation

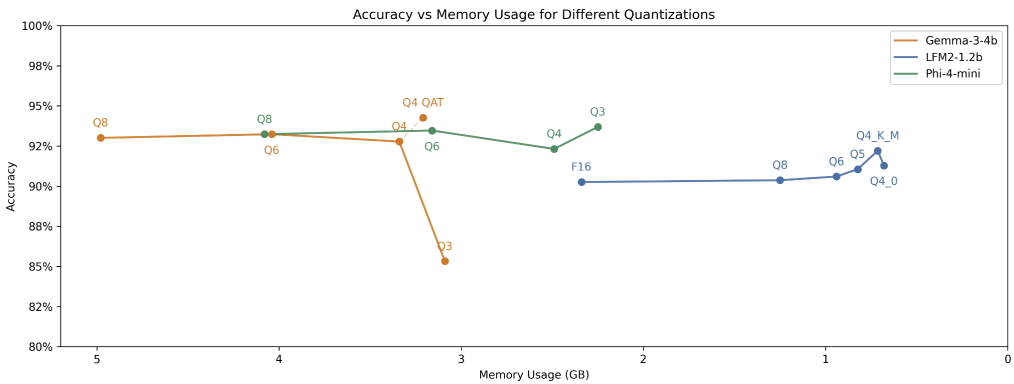

Fig. 2. Accuracy changes by quantization level for the SST-2 dataset.

4.3. Twitter Airline Sentiment Dataset

Twitter Airline results are reported as accuracy (Table 7), Macro-F1 (Table 8), and MCC (Table 9). Relative accuracy differences vs. Q8 are shown in Table 10. Figure 3 visualizes the accuracy–memory trend, and Figure 4 compares confusion matrices between Q8 and $Q4_{k\_m}$.

The Twitter Airline dataset shows largely stable performance across quantization levels, with the most notable degradation appearing for the Q3 quantization of Phi-4 Mini (Tables 7–10 and Figure 3).

To move beyond aggregate metrics, we compare confusion matrices between Q8

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.811	0.809	—	0.798	0.813	0.800
Phi 4 Mini	—	0.809	0.816	—	0.820	—	0.777
LFM2 1.2B	0.741	0.742	0.741	0.743	0.742	0.740	—

Table 7: Twitter Airline sentiment dataset accuracies

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.726	0.720	—	0.686	0.740	0.705
Phi 4 Mini	—	0.751	0.762	—	0.760	—	0.740
LFM2 1.2B	0.540	0.542	0.538	0.544	0.530	0.529	—

Table 8: Twitter Airline sentiment dataset Macro-F1 scores

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.640	0.636	—	0.617	0.646	0.611
Phi 4 Mini	—	0.653	0.661	—	0.661	—	0.621
LFM2 1.2B	0.498	0.500	0.496	0.502	0.502	0.502	—

Table 9: Twitter Airline sentiment dataset MCC scores

Model	F16	Q8	Q6	Q5	Q4	Q4/QAT	Q3
Gemma 3 4B IT	—	0.0	-0.32	—	-1.63	+0.19	-1.46
Phi 4 Mini	—	0.0	+0.83	—	+1.37	—	-3.89
LFM2 1.2B	-0.08	0.0	-0.16	+0.17	-0.04	-0.33	—

Table 10: Accuracy difference (%) vs. Q8 baseline for Twitter Airline sentiment. Positive values indicate improvement; negative values indicate degradation

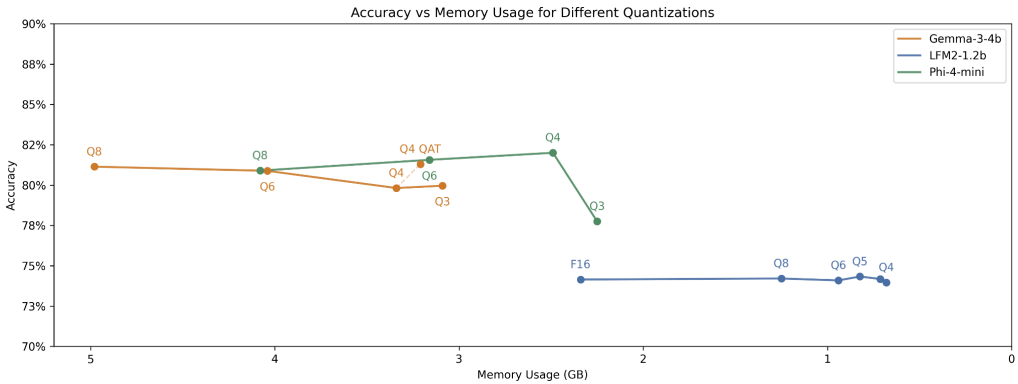

Fig. 3. Accuracy changes by quantization level for the Twitter Airline Sentiment dataset.

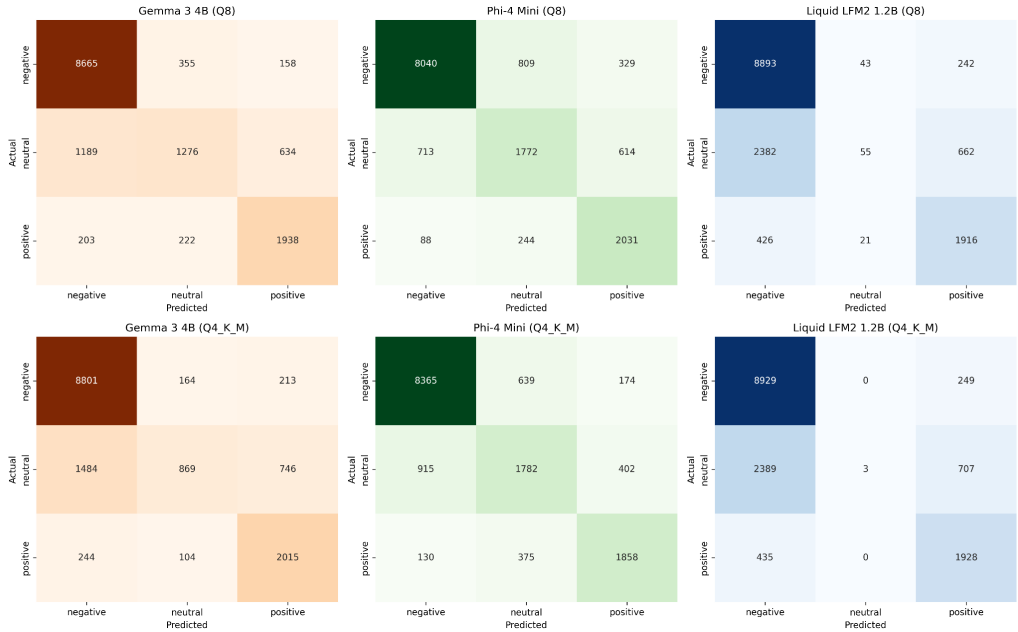

Fig. 4. Confusion matrices on the Twitter Airline Sentiment dataset for Q8 and Q4K-M quantizations of each model

and Q4K-M (Figure 4). Across all models, the Q4 quantizations increase the tendency to predict negative sentiment. For Liquid LFM2, Q4K-M largely suppresses neutral predictions, while for Gemma-3-4B a more polarized decision pattern can preserve accuracy due to the dataset's class imbalance. For Phi-4 Mini, a stronger negative bias is offset by reduced positive labeling, contributing to the observed degradation.

More broadly, these distortions are consistent with known effects of low-precision inference: reducing numerical precision in the weight matrices can amplify small variations in token-level probability distributions. In classification, this may sharpen decision boundaries such that one class absorbs disproportionate probability mass; at sufficiently low precision, rounding noise in logits can flip the top-ranked token or bias the softmax toward a dominant class.

5. Quantization Effects in Sentiment Classification

An unexpected outcome of our experiments was the minimal degradation in sentiment classification accuracy following aggressive weight quantization. Although quantization substantially reduced model file size—often by more than half—the observed decrease in prediction accuracy was consistently within a few percentage points. This section discusses why such robustness emerges, the role of temperature and output structure, and the limited runtime speed improvements seen in our experiments.

5.1. Robustness of Sentiment Classification to Quantization

Quantization perturbs the internal representations of a model, yet in classification tasks, its impact on final predictions depends heavily on logit margins. When the probability assigned to the correct class is significantly higher than alternatives, small perturbations rarely alter the top-1 prediction. Since sentiment benchmarks such as SST-2, IMDB, and Twitter Airline largely consist of unambiguous binary examples, the margin between the correct and incorrect label is typically large. Consequently, weight quantization—even to 4 bits—rarely flips predictions. This phenomenon is less pronounced in multi-class or fine-grained tasks, where decision boundaries are closer.

5.2. Temperature and Determinism

We employed a deterministic decoding setting (temperature = 0) throughout. While this does not prevent accuracy loss, it ensures that small logit perturbations manifest only if they alter the top-ranked token. In contrast, higher temperatures or stochastic sampling can amplify such perturbations over multiple generated tokens. Thus, classification scenarios with single-token outputs at temperature 0 represent a particularly stable regime for quantized inference.

5.3. Expected Accuracy Loss Across Quantization Levels

Prior work on general language tasks indicates that well-designed quantization schemes (e.g., Q6_K, Q5_K_M, Q4_K_M, or recent variants like IQ4_XS) typically incur ≤ 1% accuracy loss relative to FP16 for classification. More aggressive 3-bit quantization can degrade performance by 2–5% or more, particularly on harder or longer reasoning tasks. Our results align with this literature, confirming that sentiment classification is unusually resilient to weight compression.

5.4. Implications

The key implication is that binary sentiment classification is unusually safe to quantize: large reductions in memory footprint are achievable with negligible impact on predictive accuracy. For practitioners, this means that models like Gemma 3 4B or Phi-4 Mini can often be deployed at 4-bit precision without noticeable quality loss, making them well-suited for environments with strict efficiency or resource constraints. At the same time, the tendency of lower-precision variants to exaggerate class imbalance, especially on datasets with skewed distributions, signals an important caveat. While accuracy metrics remain stable, finer-grained behaviors—such as neutral class recognition—may be eroded. Future work should therefore expand to multi-class sentiment settings, fine-grained polarity distinctions, and longer text inputs, where quantization errors are expected to play a larger role in shaping model predictions.

6. Effects on Performance

6.1. Performance on non-native hardware

Our experiments revealed that sub-8-bit quantizations (e.g., Q6, Q5, Q4_K_M, Q3) were not faster than Q8_0 on a Radeon 890M integrated GPU, and in most cases were slightly slower. This initially appears counterintuitive: moving fewer bits through memory should, in theory, reduce latency on bandwidth-limited systems. However, the explanation lies in the absence of native low-bit execution support in many current GPUs and backends.

On platforms such as RDNA3/3.5 iGPUs accessed through the Vulkan backend, Q8_0 benefits from mature INT8 kernels that map efficiently to cooperative matrix instructions. By contrast, formats below 8 bits must first be unpacked and dequantized into higher precision tiles (typically INT8 or FP16) before multiplication. This process requires additional ALU operations, incurs extra memory traffic from per-block scale tables, and can therefore outweigh the theoretical bandwidth savings. K-style formats exacerbate this effect by storing multiple scale and offset values per block, leading to higher compute overhead despite their accuracy benefits.

We also observed that workload shape plays a major role. In short, batch-1, classificationoriented tasks such as sentiment analysis, runtime is dominated by the prefill stage and kernel launch overheads rather than long autoregressive decoding. In this regime, the relative cost of unpacking and dequantization is particularly pronounced, which explains why smaller quantizations did not translate into latency improvements. Only the simpler Q4_0 format, with its streamlined block layout and efficient kernel implementation, showed marginal gains compared to Q8_0.

These findings underscore an important distinction: file size reduction through quantization does not necessarily guarantee faster inference. Without hardware pathways for executing sub-8-bit multiplications directly, the cost of emulation in software or shader kernels can dominate. This explains why our results favored Q8_0 on non-native hardware, even though Q6 or Q4_K variants theoretically require less memory bandwidth. It also highlights the need for careful, hardware-aware evaluation: what appears to be an efficiency gain at the model level may not translate into system-level performance improvements unless the underlying processor natively supports the target precision.

Processor	Native low-bit format (s)	Potential uplift
NVIDIA B200	FP4 (NVFP4)	1.3×–1.8× vs. FP8
NVIDIA A100 / T4	INT4	up to 2× vs. INT8
Snapdragon 8 Elite 5	INT2, INT4, INT8, FP8	1.4×–2.5× (INT4), 2×–3.5× (INT2)
Intel NPU 4	INT4	1.3×–2× vs. INT8
AMD XDNA 2 NPU	INT4 / UINT4	1.3×–1.8× vs. INT8
Arm Ethos-U85	INT8 / INT16	No sub-8-bit support

Table 11: Representative processors with native low-bit quantization support, and expected performance uplift compared to INT8/FP8. Values are estimated for bandwidth-limited workloads; compute-limited layers may see smaller gains.

6.2. Native support for low precision

The performance profile of low-bit quantization changes substantially when the target hardware implements sub-8-bit arithmetic natively. In such cases, the quantized weights are not unpacked into INT8 or FP16 tiles before multiplication but are instead consumed directly by specialized tensor cores or NPUs. This distinction is crucial: when low-bit multiplications occur in hardware, the theoretical advantages of reduced memory bandwidth and storage are translated into actual improvements in latency and energy efficiency. By contrast, on platforms without such support, the need to dequantize and apply per-block scaling factors erodes most of the potential gains, as observed in our experiments with Q6, Q5, and Q4_K variants.

Native execution pathways therefore determine whether smaller quantizations provide real acceleration or remain largely a storage optimization. Modern processors have begun to introduce such support at scale. NVIDIA's Blackwell architecture, for instance, adds native FP4 tensor operations; Qualcomm's Snapdragon 8 Elite Gen 5 enables INT2 and INT4 execution on its Hexagon NPU for edge workloads; Intel's Core Ultra ("Lunar Lake" and "Meteor Lake") incorporates INT4 support into its on-die NPUs; and AMD's latest Ryzen AI 300-series integrates INT4/UINT4 pipelines via XDNA 2. These examples show that hardware vendors are actively designing accelerators around sub-8-bit arithmetic, making it increasingly relevant for deployment-oriented research.

The following subsection provides a detailed survey of representative processors, the formats they natively support, and the expected performance uplifts compared to INT8 or FP8 baselines under bandwidth-limited conditions.

6.3. Processor with native sub-8-bit support

The following processors implement true sub-8-bit execution paths, allowing quantized models to achieve actual runtime gains rather than incurring dequantization overhead:

- NVIDIA Blackwell (B200 / GB200; "RTX Blackwell" derivatives). Native format: FP4 via the Transformer Engine (NVFP4). Supports hardware-accelerated 4-bit floating-point matrix operations with micro-scaled FP8/FP32 factors. Memory shrinkage is about 1.8× compared to FP8 (and about 3.5× compared to FP16), giving 1.3×–1.8× throughput gains in bandwidth-dominated layers. (NVIDIA, 2024)
- NVIDIA Ampere / Turing (A100, T4, RTX Turing). Native format: INT4 (and INT8) Tensor Cores. Architecture papers show INT4 throughput is 2× INT8 (32× vs. 16× FP32), and A100 doubles Turing's Tensor Core throughput. With INT4 kernels, up to 2× speedup over INT8 is achievable when bandwidth dominates. (NVIDIA, 2020)(NVIDIA, 2018)
- Qualcomm Snapdragon 8 Elite Gen 5 (SM8850-AC; Hexagon NPU). Native formats: INT2, INT4, INT8, INT16, FP8, FP16. INT2/INT4 execution reduces weight bandwidth by factors of 4× and 2× respectively compared to INT8. For small to medium LLMs on the NPU, this corresponds to 1.4×–2.5× (INT4) and 2×–3.5× (INT2) throughput gains in prefill-bound workloads, with additional power savings. (Qualcomm, 2024)

- Intel Core Ultra ("Lunar Lake", "Meteor Lake") with NPU 4. Native format: INT4, exposed via the Intel NPU Acceleration Library v1.2. Models compiled to true INT4 NPU ops can expect 1.3×–2× uplift over INT8 in bandwidth-limited attention/MLP layers. (Intel, 2024)
- AMD Ryzen AI 300-series with XDNA 2 NPU. Native/workflow formats: INT4 / UINT4, exposed through Ryzen AI Software, Quark, and ONNX Runtime GenAI. Weight-only INT4 reduces bandwidth by about 2× compared to INT8, giving 1.3×–1.8× gains in prefill-dominated settings. On discrete AMD GPUs/CDNA, native support is limited to FP8/INT8, not FP4. (AMD, 2024)
- Arm Ethos-U85 (micro-NPU). Native formats: INT8 weights and INT8/INT16 activations. There is no public claim of INT4 or FP4 execution, making this a useful counterexample that not every NPU exposes sub-8-bit support today. (Arm, 2023)

6.4. Implications for deployment

The distinction between non-native and native execution of sub-8-bit arithmetic has direct consequences for deployment strategies. On current consumer GPUs, our results showed that formats below INT8 may shrink model size but do not necessarily improve runtime performance. The absence of true low-bit tensor operations means that the cost of unpacking, dequantization, and scale application can dominate execution time, often making Q8_0 the most efficient choice in practice. In such environments, sub-8-bit quantization primarily provides storage and VRAM savings rather than speed gains.

The picture is very different on processors that natively implement INT4 or even INT2 arithmetic. In these cases, the multiply–accumulate operations are executed directly at the low precision, and the theoretical reductions in memory bandwidth translate into real latency and throughput improvements. For small LLMs that already retain accuracy under aggressive quantization, this enables both faster inference and lower energy consumption.

Among current hardware platforms, the Qualcomm Snapdragon 8 Elite Gen 5 stands out as particularly relevant. Its Hexagon NPU supports INT2 and INT4 execution, allowing weight bandwidth to be reduced by factors of four and two compared to INT8. For compact sentiment analysis models in the 1–4B parameter range, this can deliver measurable performance gains—roughly 1.4×–2.5× in INT4 and up to 2×–3.5× in INT2 modes under bandwidth-limited conditions—while simultaneously reducing power draw. This combination makes real-time, on-device inference feasible for tasks such as customer feedback monitoring, content moderation, or personalized recommendations directly on smartphones.

The implications for deployment are therefore significant. With native INT2/INT4 support in mobile NPUs, aggressive quantization becomes a true performance optimization rather than just a memory-saving measure. For small models, which are already light enough to fit on-device, the availability of such hardware means that quantized sentiment analysis can run privately, efficiently, and at scale on consumer phones. This reduces reliance on cloud APIs, cuts costs, and strengthens data privacy, positioning edge devices as the natural target platform for compact quantized LLMs.

7. Discussion

The findings of this study demonstrate that sentiment classification tasks are notably robust to aggressive quantization, with most models retaining accuracy within a narrow margin of their Q8 baselines even at 4-bit precision. This resilience can be attributed to the relatively clear decision boundaries in binary sentiment classification and the deterministic decoding strategy adopted in our experiments. However, the results also reveal nuanced shifts in model behavior, particularly on imbalanced datasets, where low-bit quantization tends to amplify class polarization and reduce the ability to recognize minority classes such as “neutral.”

Across datasets, Gemma 3 4B exhibited the most stable performance under quantization, with its QAT-trained variant even surpassing the Q8 baseline on IMDB and SST-2. This underscores the potential of quantization-aware training to mitigate accuracy loss and highlights the value of integrating training-time adaptations when aggressive compression is required. Phi-4 Mini similarly maintained high accuracy at 6- and 5-bit quantizations but showed evidence of prediction skew under Q4_K_M, suggesting that compact models may be more sensitive to distributional shifts introduced by reduced precision. Liquid LFM2, despite its smaller size, achieved stable results under 4-bit quantization, though its handling of nuanced or imbalanced inputs degraded more visibly than its larger counterparts.

From a performance perspective, the experiments confirm that file-size reductions do not necessarily translate into runtime speed gains on non-native hardware. In fact, sub-8-bit quantizations frequently incurred additional unpacking and dequantization overhead, resulting in equal or slower inference compared to Q8_0 on GPUs lacking native support. This aligns with recent literature emphasizing that the benefits of quantization are contingent on hardware execution pathways. Where native support exists—such as in the Snapdragon 8 Elite Gen 5 with INT2/INT4, NVIDIA Blackwell with FP4, or Intel’s and AMD’s NPUs with INT4—quantization delivers real improvements in both speed and efficiency. These results reinforce the need for hardware-aware evaluations: aggressive quantization may be a practical storage optimization on legacy systems, but it only becomes a performance optimization on modern processors with dedicated low-bit tensor cores or NPUs.

Taken together, these observations suggest several directions for future work. First, evaluations should extend to multi-class and fine-grained sentiment datasets, where quantization-induced distortions are likely to manifest more strongly. Second, systematic comparisons between post-training quantization and QAT across model families are needed to clarify the conditions under which training-time adaptations yield consistent benefits. Finally, bridging the gap between software toolchains and emerging hardware capabilities will be essential to fully realize the advantages of low-bit quantization. As native INT2/INT4 support proliferates in consumer and edge devices, quantized small and mid-sized LLMs stand to become the default choice for real-time, privacy-preserving sentiment analysis at scale.

8. Conclusion

This work systematically evaluated the effects of post-training quantization on three representative large language models for sentiment classification: Gemma 3 4B, Microsoft Phi-4 Mini, and Liquid LFM2 1.2B. By comparing multiple quantization levels—ranging from FP16 and Q8 to more aggressive schemes such as Q4, Q4-QAT, and Q3—we demonstrated that sentiment analysis is unusually resilient to weight compression. Across IMDB, SST-2, and Twitter Airline datasets, accuracy reductions relative to Q8 were generally limited to a few percentage points, with the most substantial declines only appearing under 3-bit quantization. Notably, Gemma's QAT-trained 4-bit variant even outperformed its higher-precision baseline, underscoring the promise of training-time adaptation in pre-serving quality.

At the same time, our analysis revealed behavioral shifts introduced by lower-precision formats. On imbalanced datasets, quantization amplified class polarization, often diminishing the recognition of minority labels such as "neutral." These distortions are important to consider in real-world deployments, where fairness and consistency may matter as much as raw accuracy.

From a systems perspective, we found that quantization primarily yields storage and memory benefits on current GPUs without native sub-8-bit execution, but does not always translate into runtime gains. In contrast, hardware with built-in support for INT4 or INT2 arithmetic—such as Qualcomm's Snapdragon 8 Elite Gen 5, NVIDIA Blackwell, or Intel and AMD NPUs—can transform quantization into a true performance optimization, enabling faster, more energy-efficient inference.

Overall, the results highlight both the opportunities and caveats of quantization for sentiment analysis. Compact LLMs at 4-bit precision already achieve a favorable balance between efficiency and accuracy, making them strong candidates for on-device or resource-constrained applications. As hardware ecosystems increasingly embrace native low-bit execution, the deployment of quantized models at scale will become not just feasible but advantageous, supporting real-time, private, and cost-effective sentiment analysis across diverse domains.

References

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv preprint arXiv:1510.00149.

Team, G., Kamath, A., Ferret, J., Pathak, S., Vieillard, N., Merhej, R., Perrin, S., Matejovicova, T., Ramé, A., Rivière, M., et al. (2025). Gemma 3 technical report. arXiv preprint arXiv:2503.19786.

Abdin, M., Aneja, J., Behl, H., Bubeck, S., Eldan, R., Gunasekar, S., Harrison, M.,

Hewett, R. J., Javaheripi, M., Kauffmann, P., et al. (2024). Phi-4 technical report. arXiv preprint arXiv:2412.08905.

Liquid AI. (2025). Introducing LFM2: The fastest on-device foundation models on the market. https://www.liquid.ai/blog/liquid-foundation-models-v2-our-second-series-of-generative-ai-models

Zafrir, O., Boudoukh, G., Izsak, P., & Wasserblat, M. (2019). Q8BERT: Quantized 8-bit BERT. IEEE.

Shen, S., Dong, Z., Ye, J., Ma, L., Yao, Z., Gholami, A., Mahoney, M. W., & Keutzer, K. (2020). Q-BERT: Hessian based ultra low precision quantization of BERT. AAAI.

Frantar, E., Ashkboos, S., Hoefler, T., & Alistarh, D. (2022). GPTQ: Accurate post-training quantization for generative pre-trained transformers. arXiv.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., & Han, S. (2023). SmoothQuant. ICML.

Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.-M., Wang, W.-C., Xiao, G., Dang, X., Gan, C., & Han, S. (2024). AWQ. Proceedings of Machine Learning and Systems.

Dettmers, T., Svirschevski, R., Egiazarian, V., Kuznedelev, D., Frantar, E., Ashkboos, S., Borzunov, A., Hoefler, T., & Alistarh, D. (2023). SpQR. arXiv.

Kim, S., Hooper, C., Gholami, A., Dong, Z., Li, X., Shen, S., Mahoney, M. W., & Keutzer, K. (2023). SqueezeLLM. arXiv.

Zhu, X., Li, J., Liu, Y., Ma, C., & Wang, W. (2024). A survey on model compression for large language models. TACL.

Jin, R., Du, J., Huang, W., Liu, W., Luan, J., Wang, B., & Xiong, D. (2024). Evaluation of quantization strategies. ACL Findings.

Li, S., Ning, X., Wang, L., Liu, T., Shi, X., Yan, S., Dai, G., Yang, H., & Wang, Y. (2024). Evaluating quantized LLMs. arXiv.

Zhang, T., Yi, J., Xu, Z., & Shrivastava, A. (2024). KV cache quantization. NeurIPS.

Hooper, C., Kim, S., Mohammadzadeh, H., Mahoney, M. W., Shao, Y. S., Keutzer, K., & Gholami, A. (2024). KVQuant. NeurIPS.

Dettmers, T., Lewis, M., Belkada, Y., & Zettlemoyer, L. (2022). LLM.int8(). NeurIPS.

Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). QLoRA. NeurIPS.

Hansen, C., et al. (2023). AutoAWQ. https://github.com/casper-hansen/AutoAWQ

Matthews, B. W. (1975). Comparison of predicted and observed structure. Biochimica et Biophysica Acta.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2018). GLUE benchmark. arXiv.

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. ACL.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive deep models. EMNLP.Crowdflower. (2015).

Twitter US airline sentiment dataset. https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment

NVIDIA. (2024). NVIDIA Blackwell architecture in-depth. https://www.nvidia.com/en-us/data-center/blackwell-architecture/

NVIDIA. (2020). NVIDIA A100 tensor core GPU architecture. https://www.nvidia.com/en-us/data-center/a100/

NVIDIA. (2018). NVIDIA Turing architecture whitepaper. https://www.nvidia.com/en-us/data-center/turing-architecture/

Qualcomm. (2024). Snapdragon 8 Elite Gen 5 mobile platform product brief. https://www.qualcomm.com/products/mobile/snapdragon-8-elite-gen-5

Intel. (2024). Intel NPU acceleration library v1.2 release notes. https://github.com/intel/neural-compressor/releases

AMD. (2024). Ryzen AI 300 series with XDNA 2: Developer guide. https://www.amd.com/en/products/ryzen-ai

Arm. (2023). Arm Ethos-U85 NPU product brief. https://www.arm.com/products/silicon-ip-npu/ethos-u85

Submitted: 25.12.2025
Accepted: 10.04.2026