"Give Me BF16 or Give Me Death"? Accuracy-Performance Trade-Offs in LLM Quantization

Eldar Kurtić[1,2], Alexandre Marques[1], Shubhra Pandit[1], Mark Kurtz[1], Dan Alistarh[1,2]

[1]Red Hat AI, [2]Institute of Science and Technology Austria

Correspondence: ekurtic@redhat.com, dalistar@redhat.com

Abstract

Quantization is a powerful tool for accelerating large language model (LLM) inference, but the accuracy-performance trade-offs across different formats remain unclear. In this paper, we conduct the most comprehensive empirical study to date, evaluating FP8, INT8, and INT4 quantization across academic benchmarks and real-world tasks on the entire Llama-3.1 model family. Through over 500,000 evaluations, our investigation yields several key findings: (1) FP8 (W8A8-FP) is effectively lossless across all model scales, (2) well-tuned INT8 (W8A8-INT) achieves surprisingly low (1-3%) accuracy degradation, and (3) INT4 weight-only (W4A16-INT) is more competitive than expected, rivaling 8-bit quantization. Further, we investigate the *optimal* quantization format for different deployments by analyzing inference performance through the popular vLLM framework. Our analysis provides clear deployment recommendations: W4A16 is the most cost-efficient for synchronous setups, while W8A8 dominates in asynchronous continuous batching. For mixed workloads, the optimal choice depends on the specific use case. Our findings offer practical, data-driven guidelines for deploying quantized LLMs at scale—ensuring the best balance between speed, efficiency, and accuracy.

1 Introduction

The high computational cost of serving LLMs has driven extensive research into inference acceleration techniques, including quantization (Frantar et al., 2022; Dettmers and Zettlemoyer, 2022; Lin et al., 2024a), speculative decoding (Chen et al., 2023; Leviathan et al., 2023), and pruning (Xia et al., 2023; Muralidharan et al., 2024). Among these, quantization—reducing the bitwidth of weights, activations, or both—has emerged as the most widely used approach. However, its key challenge lies in balancing efficiency and accuracy. Despite progress, systematic benchmarks and practical deployment guidelines remain scarce. This uncertainty has fueled speculation around quantized models, exemplified by the initial skepticism toward the Llama-3.1-405B quantized model release (Dubey et al., 2024), which was later found to be near-lossless in LMSYS Arena user evaluations (Chiang et al., 2024). To address this gap, we pose the following core question:

What are the practical accuracy-performance trade-offs for popular quantization formats?

In this study, we focus on widely supported, computationally efficient quantization formats. Specifically, we examine 8-bit weights and activations (W8A8), using integer (INT) precision for NVIDIA Ampere and older GPUs and floating-point (FP) precision for NVIDIA Hopper and Ada Lovelace. Additionally, we consider 4-bit integer weights with 16-bit activations (W4A16-INT), a competitive low-bit alternative. To evaluate accuracy, we implement a broad automated evaluation suite, spanning both academic and real-world benchmarks. Our academic benchmarks include Open LLM Leaderboard V1 (Beeching et al., 2023) and its more challenging V2 version (Fourrier et al., 2024), while real-world generative tasks are represented by Arena-Hard-Auto-v0.1 (Li et al., 2024b), HumanEval (Chen et al., 2021) and HumanEval+ (Liu et al., 2023a), and the long-context RULER benchmark (Hsieh et al., 2024). Beyond standard evaluations, we further analyze text similarity between outputs from uncompressed and quantized models to assess generative consistency. Finally, we conduct an extensive inference performance study, benchmarking vLLM (Kwon et al., 2023) (version 0.6.4.post1) across three GPU architectures (A6000, A100, H100) in seven deployment scenarios. Our findings provide a comprehensive view of quantization's trade-offs and offer practical recommendations for real-world LLM deployment. Our main findings are as follows:

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26872–26886

1. **W8A8-FP quantization is essentially lossless**, preserving the uncompressed model's accuracy across all benchmarks, often within the evaluation's margin of error. This result is achieved with a simple yet robust approach: dynamic per-token activation quantization combined with symmetric weight quantization via round-to-nearest assignment.

2. **W8A8-INT quantization exhibits only a modest accuracy degradation** of 1–3% per task on average, far lower than the 10%+ drops reported in prior work (Li et al., 2024a; Lee et al., 2024b). This performance is enabled by dynamic activation quantization or SmoothQuant (Xiao et al., 2022), paired with GPTQ (Frantar et al., 2022) for symmetric weight quantization.

3. **W4A16-INT quantization maintains consistently low accuracy loss, performing on par with W8A8-INT**. Surprisingly, we show for the first time that a simple variant of GPTQ outperforms the more recent AWQ method (Lin et al., 2024a) on real-world tasks, challenging prior assumptions about low-bit quantization strategies.

4. **Beyond accuracy, our text similarity analysis reveals that larger quantized models closely adhere to the word choices and sentence structures of their uncompressed counterparts in autoregressive text generation**. In contrast, smaller quantized models introduce moderate variability in structure but still preserve semantic meaning.

5. **In terms of performance, W4A16-INT is the most efficient choice for synchronous deployments, while W8A8 formats maximize throughput in asynchronous settings.** The optimal quantization scheme depends on model size, hardware, and deployment needs—whether for latency-sensitive applications like code completion or high-throughput multi-turn chat.

Overall, this work provides the first in-depth study of accuracy vs. performance vs. cost trade-offs for quantized LLMs across formats, algorithms, use cases, and hardware types. We aim for these findings to serve as both a practical deployment guide and a strong and competitive foundation for future research on better quantization techniques.

2 Background and Related Work

2.1 A Primer on Quantization

Early work focused on INT8 activation quantization and INT4/INT8 weight quantization (Dettmers et al., 2022; Yao et al., 2022; Park et al., 2022). A common approach is round-to-nearest (RTN) over groups: given a group of g consecutive weights as a vector $\mathbf{x} \in \mathbb{R}^g$, b-bit RTN is defined as:

$$\begin{aligned} \mathcal{Q}(\mathbf{x}, b) &= \mathrm{rnd}\left(\frac{\mathbf{x} - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})}(2^b - 1)\right) \\ &= \mathrm{rnd}((\mathbf{x} - z(\mathbf{x}))/s(\mathbf{x})), \end{aligned} \tag{1}$$

where rnd rounds to the nearest integer, $z(\mathbf{x}) = \min(\mathbf{x})$ is the zero point, and $s(\mathbf{x}) = (\max(\mathbf{x}) - \min(\mathbf{x}))/(2^b - 1)$ is the scale, computed using min-max normalization. However, RTN struggles at INT4 precision and suffers from lossy activation quantization even at INT8 (Dettmers et al., 2022).

Weight Quantization. To mitigate weight quantization errors, GPTQ (Frantar et al., 2022) introduced second-order weight adjustments using calibration data. Subsequent methods, including AWQ (Lin et al., 2024a), SqueezeLLM (Kim et al., 2023), OWQ (Lee et al., 2024a), and SpQR (Dettmers et al., 2023), incorporated outlier-aware quantization, storing a fraction of weights in higher precision to enable highly accurate 4-bit quantization. More recent high-compression techniques—QuIP (Chee et al., 2023), QuIP# (Tseng et al., 2024a), QTIP (Tseng et al., 2024b), AQLM (Egiazarian et al., 2024), and GPTVQ (van Baalen et al., 2024)—target low-bitwidths using advanced representations such as vector quantization. Yet, these formats are inefficient for batch sizes larger than 1, limiting their practicality.

Activation Quantization. Quantizing both weights and activations enables low-bit hardware operations. Yet, activations are difficult to quantize due to *outlier features*—elements up to 100× larger than the average (Dettmers et al., 2022). Early attempts extracted outlier columns at runtime, but this is inefficient. SmoothQuant (Xiao et al., 2022) improves upon this by noticing that outliers are stable across the model and can be precomputed using a calibration set. Follow-up work explored W4A4 quantization (Ashkboos et al., 2023, 2024) and mixed-precision W4A8 (Lin et al., 2024b; Zhang et al., 2024), including KV-cache quantization. While promising, these methods still suffer accuracy loss and lack robust support in high-performance inference frameworks.

2.2 Related Work

A significant body of work has explored the accuracy trade-offs under different quantization schemes (Yao et al., 2023; Liu et al., 2023b; Huang et al., 2024; Gong et al., 2024b; Li et al., 2024a; Gong et al., 2024a). However, much of this research relies primarily on academic benchmarks, which do not fully reflect real-world deployment scenarios. Additionally, the lack of hyperparameter tuning in some studies leads to misleading conclusions about accuracy, as we demonstrate in our experiments. We challenge the claim that 8-bit integer activation quantization causes substantial accuracy degradation (Li et al., 2024a; Lee et al., 2024b), providing vast evidence to the contrary.

The closest work to ours is by Lee et al. (2024b), which, like most prior studies, focuses on *quantization accuracy*, but overlooks key factors. First, while the authors claim to analyze models up to 405B parameters, they omit open-ended benchmarks at this scale and fail to report full-precision baselines even for academic tasks. Without these references, the impact of quantization remains unclear. To address this, we enable efficient multi-node evaluations for the 405B model, conducting a comprehensive accuracy analysis in both academic and real-world settings. Second, Lee et al. (2024b) asserts that AWQ outperforms GPTQ in a 4-bit weight-only quantization setup. We correct this claim, and attribute it to suboptimal hyperparameter choices. Our comparative analysis (Table 1 and Appendix A.2) shows that while both methods perform similarly on academic benchmarks, GPTQ exhibits notable gains over AWQ in real-world tasks, particularly coding.

Third, we refute the conclusion that W8A8-INT is significantly inferior to W8A8-FP and W4A16-INT. With proper tuning, W8A8-INT achieves competitive accuracy, with only minor losses. For example, while Lee et al. (2024b) reports a 10-point accuracy drop for W8A8-INT quantized 405B models on the Open LLM Leaderboard V2 compared to FP8, our approach reduces this to just 0.7 points.

3 Benchmark Design and Setup

3.1 Datasets and Benchmarks

We categorize benchmarks into three groups: academic, real-world, and text similarity analysis.

1. Academic benchmarks, such as Open LLM Leaderboard V1 and V2 (Beeching et al., 2023; Fourrier et al., 2024), provide structured evaluations for question-answering and reasoning tasks. While widely used for benchmarking, they lack alignment with real-world scenarios involving semantics, variability, and context-awareness. Leaderboard V1 includes tasks like GSM for grade school math (Cobbe et al., 2021), MMLU and ARC-Challenge for world knowledge and reasoning (Hendrycks et al., 2020; Clark et al., 2018), Winogrande and HellaSwag for language understanding (Sakaguchi et al., 2021; Zellers et al., 2019), and TruthfulQA for factual correctness (Lin et al., 2021). Leaderboard V2 extends this with expert knowledge benchmarks such as MMLU-Pro (Wang et al., 2024), GPQA (Rein et al., 2023), and Big Bench Hard (Suzgun et al., 2022), as well as multi-step reasoning (MuSR (Sprague et al., 2024)), advanced math (MATH Level 5 (Hendrycks et al., 2021)), and instruction following (IFEval (Zhou et al., 2023)). By evaluating across both leaderboards, we capture a broad spectrum of reasoning and knowledge domains, using both log-likelihood and text-generation evaluations to stress-test quantized models.

2. Real-world benchmarks evaluate models in practical scenarios such as instruction following, chat, long-context, and code generation. Arena-Hard-Auto-v0.1 (Li et al., 2024b; Chiang et al., 2024; Li et al., 2024c) automates LMSYS Chatbot Arena (Chiang et al., 2024) evaluations, using an LLM to judge responses to 500 complex prompts, achieving an 89% agreement with human rankings (Li et al., 2024c). This allows rapid and scalable assessment of chat capabilities without human intervention. For code generation, we evaluate models on HumanEval (Chen et al., 2021) and its extension HumanEval+ (Liu et al., 2023a), which test the ability to generate correct and functional code. Finally, we conduct long-context evaluations via the rigorous RULER benchmark (Hsieh et al., 2024) which consists of retrieval, multi-hop tracing, information aggregation, and question answering evaluations at sequence lengths from 4k to 128k.

3. Our text similarity analysis benchmark assesses how closely quantized models' outputs align with their full-precision counterparts. While real-world benchmarks reflect practical usage, their open-ended nature introduces variability, making direct accuracy comparisons challenging. To mitigate this, we analyze output similarity under identical prompts using ROUGE (Lin, 2004), BERTScore (Zhang et al., 2019), and Semantic Textual Similarity (STS) (Reimers and Gurevych, 2019). ROUGE-1 measures unigram overlap, while

Table 1: Comparison of GPTQ and AWQ 4-bit weight quantization algorithms (W4A16-INT). We observe a small gap between methods on academic benchmarks (left) but a more pronounced difference in favor of GPTQ on real-world (open-ended) benchmarks (right).

	Academic Benchmarks			Real-World Benchmarks			
Model	Average Score	Leaderboard V1	Leaderboard V2	Average Score	Arena-Hard	HumanEval	MBPP
Llama-3.1-8B-Instruct	50.84	74.06	27.62	53.7	25.8	67.3	68.1
GPTQ (Frantar et al., 2022)	49.82	**73.11**	26.53	**52.3**	**24.0**	**67.1**	**65.8**
AWQ (Lin et al., 2024a)	**50.05**	72.69	**27.40**	49.4	22.3	63.0	62.8
Llama-3.1-70B-Instruct	62.93	84.20	41.66	73.1	57.0	79.7	82.5
GPTQ (Frantar et al., 2022)	62.18	83.77	40.58	**73.1**	**57.0**	**80.5**	**81.9**
AWQ (Lin et al., 2024a)	**62.53**	**83.96**	**41.09**	72.3	56.7	79.4	80.8

ROUGE-L captures structural similarity through the longest common subsequence. BERTScore computes token-level contextual similarity using RoBERTa-large embeddings, and STS assesses semantic alignment at the sentence level via Sentence Transformers built on MiniLM (Wang et al., 2020).

3.2 Models, Formats, and Algorithms

We evaluate using the highly-popular Llama 3.1 model series (Dubey et al., 2024). To assess quantization trade-offs, we conduct experiments on the instruction-tuned versions of all available sizes (8B, 70B, and 405B). For each, we examine the three main formats with kernel support in vLLM: W8A8-FP, W8A8-INT, and W4A16-INT.

W8A8-FP quantizes all linear operators in transformer blocks to an 8-bit floating-point format, using round-to-nearest quantization. Weights follow a symmetric per-output-channel scheme, while activations are dynamically quantized per token. This requires no calibration data and remains computationally efficient, even for large-scale models.

W8A8-INT reduces weights and activations to 8-bit integers, applying symmetric per-output-channel GPTQ quantization for weights and dynamic per-token quantization for activations. While this scheme performs well for 8B and 405B models, it causes noticeable accuracy drops at 70B. To mitigate this, we apply SmoothQuant, shifting some activation complexity onto weights, which are easier to quantize. For calibration, random tokens suffice at 8B, but larger models require higher-quality calibration data, for which we use Lee et al. (2023).

W4A16-INT quantizes weights to 4-bit integers while keeping activations at 16-bit precision. Weights are compressed using GPTQ with MSE-optimal clipping, applied in 128-element groups. Unlike higher-bit formats, random token calibration degrades accuracy, so we rely on OpenPlatypus data for calibration.

INT4 Quantization Algorithms. We focus on two inference-efficient techniques: AWQ and GPTQ, evaluating them on Leaderboard V1/V2, Arena-Hard, HumanEval, and MBPP. Results (Table 1) show near-identical performance on academic benchmarks, with AWQ leading by just 0.23 and 0.35 points on a 0–100 scale. However, GPTQ outperforms AWQ on real-world tasks by wider margins (2.9 and 0.8 points, respectively), leading us to adopt GPTQ as our primary INT4 method. This finding contrasts with prior studies (Lin et al., 2024a; Huang et al., 2024), which favored AWQ or found it tied on academic subsets. We attribute this to three key factors: (1) we use GPTQ with MSE-optimal clipping (the AWQ comparison used abs-max); this has no overhead and yields consistently better results; (2) we use higher-quality calibration data than the C4 default; (3) we include real-world benchmarks, providing a broader evaluation scope.

4 Quantization Impact on Accuracy

We begin our discussion of the results by examining the accuracy of quantized models across Leaderboard V1 (Table 2), Leaderboard V2 (Table 3) and real-world benchmarks (Table 3). Given the density of the results, we discuss them individually via average recoveries across higher-level benchmarks and discuss "outlier" observations.

4.1 Academic Benchmarks

Our first analysis focuses on Open LLM Leaderboard V1 and V2, ensuring generalization by optimizing quantization hyperparameters on V1 while validating results on V2.

The Open LLM Leaderboard V1 follows Meta's prompt guidelines for Llama-3.1 models to maintain alignment with baseline scores. This introduces two key differences from standard evaluation protocols: MMLU and ARC-Challenge are assessed as text-generation tasks rather than log-likelihood-based evaluations (Gao et al., 2021), and GSM8k is tested using chain-of-thought prompting

Table 2: Detailed per-task breakdown of accuracy on a subset of academic benchmarks (Open LLM Leaderboard V1) for quantized Llama-3.1-Instruct models across all three model sizes (8B, 70B, 405B). Higher score is better.

		Recovery %	Average Score	MMLU 5-shot	MMLU CoT 0-shot	ARC-C 0-shot	GSM8k CoT 8-shot	HellaSwag 10-shot	Winogrande 5-shot	TruthfulQA 0-shot
8B	BF16	100.00	74.06	68.3	72.8	81.4	82.8	80.5	78.1	54.5
	W8A8-FP	99.31	73.55	68.0	71.6	81.2	82.0	80.0	77.7	54.3
	W8A8-INT	100.31	74.29	67.8	72.2	81.7	84.8	80.3	78.5	54.7
	W4A16-INT	98.72	73.11	66.9	71.1	80.2	82.9	79.9	78.0	52.8
70B	BF16	100.00	84.40	83.8	86.0	93.3	94.9	86.8	85.3	60.7
	W8A8-FP	99.72	84.16	83.8	85.5	93.5	94.5	86.6	84.6	60.6
	W8A8-INT	99.87	84.29	83.7	85.8	93.1	94.2	86.7	85.1	61.4
	W4A16-INT	99.53	84.00	83.6	85.6	92.8	94.4	86.3	85.5	59.8
405B	BF16	100.00	86.79	87.4	88.1	95.0	96.0	88.5	87.2	65.3
	W8A8-FP	100.12	86.89	87.5	88.1	95.0	95.8	88.5	88.0	65.3
	W8A8-INT	99.32	86.20	87.1	87.7	94.4	95.5	88.2	86.1	64.4
	W4A16-INT	99.98	86.78	87.2	87.7	95.3	96.3	88.3	87.4	65.3

Table 3: Detailed per-task breakdown of accuracy on a subset of academic (Open LLM Leaderboard V2) and on real-world (Arena-Hard, HumanEval, RULER) benchmarks for quantized Llama-3.1-Instruct models across all three model sizes (8B, 70B, 405B). Higher score is better. Long-context RULER evaluations at 405B are prohibitively expensive for our cluster.

		Academic Benchmarks (Open LLM Leaderboard V2)								Real-World Benchmarks			
		Recovery %	Average Score	IFEval 0-shot	BBH 3-shot	Math lvl 5 4-shot	GPQA 0-shot	MuSR 0-shot	MMLU-Pro 5-shot	Arena-Hard Win-Rate	HumanEval pass@1	HumanEval+ pass@1	RULER Score
8B	BF16	100.0	27.6	77.8	30.1	15.7	3.7	7.6	30.8	25.8	67.3	60.7	82.8
	W8A8-FP	101.2	27.9	77.2	29.6	16.5	5.7	7.5	31.2	26.8	67.3	61.3	82.8
	W8A8-INT	101.5	28.0	77.9	30.9	15.5	5.4	7.6	30.9	27.2	67.1	60.0	82.8
	W4A16-INT	96.1	26.5	76.3	28.9	14.8	4.1	6.3	28.8	24.0	67.1	59.1	81.1
70B	BF16	100.0	41.7	86.4	55.8	26.1	15.4	18.1	48.1	57.0	79.7	74.8	83.3
	W8A8-FP	100.0	41.7	87.6	54.9	28.0	14.6	17.2	47.7	57.7	80.0	75.0	83.0
	W8A8-INT	97.3	40.5	86.6	55.2	23.9	13.6	16.8	47.1	57.0	78.7	74.0	82.5
	W4A16-INT	97.4	40.6	85.7	55.0	24.4	13.8	17.2	47.2	56.3	80.5	74.2	82.2
405B	BF16	100.0	48.7	87.7	67.0	38.9	19.5	19.5	59.7	67.4	86.8	80.1	-
	W8A8-FP	99.9	48.7	86.8	67.1	38.8	18.9	20.8	59.4	66.9	87.0	81.0	-
	W8A8-INT	98.3	47.9	86.9	66.7	35.8	20.4	19.2	58.4	64.6	86.9	80.4	-
	W4A16-INT	98.9	48.2	88.0	67.5	37.6	17.5	19.4	59.3	66.5	85.1	78.9	-

instead of a few-shot approach.

Table 2 shows that all quantization schemes, across model sizes, recover approximately 99% of the unquantized BF16 baseline. The lowest task recovery occurs on TruthfulQA, reaching 96.88% for W4A16-INT at 8B and ∼98.5% for larger models (see Appendix Table 10). On average, 8-bit quantization achieves 99.75% recovery, while W4A16-INT reaches a competitive 99.36%.

The Open LLM Leaderboard V2 incorporates more challenging tasks to assess advanced reasoning. Unlike V1, V2 normalizes scores by subtracting the random baseline and rescaling to a 0-100 range, ensuring equal weighting across tasks regardless of inherent difficulty.

Table 3 shows that quantized models maintain 99% of the baseline's average score, with all models recovering at least 96%. However, due to the increased difficulty, smaller models exhibit higher variance, particularly on GPQA and MuSR, where full-precision models already approach random guessing thresholds, reducing the reliability of accuracy recovery signals (Appendix Table11). Focusing on tasks where the full-precision model scores above 40%, ensuring a meaningful performance baseline, we observe the lowest per-task recovery for 8-bit FP quantization at 98.44% on BBH (70B) and for 8-bit INT at 97.8% on MMLU-Pro (405B). Notably, W4A16-INT models demonstrate superior recovery over W8A8-INT, with a minimum accuracy retention of 98% for the 8B model on IFEval. This suggests that, for INT, quantizing activations is harder than quantizing weights.

4.2 Real-World Benchmarks

While academic benchmarks offer structured evaluations, real-world benchmarks better capture model performance in dynamic environments. These evaluations involve diverse prompts, longer generations, and multiple valid responses, emphasizing correctness and semantic quality. We assess four key benchmarks: Arena-Hard-Auto-v0.1

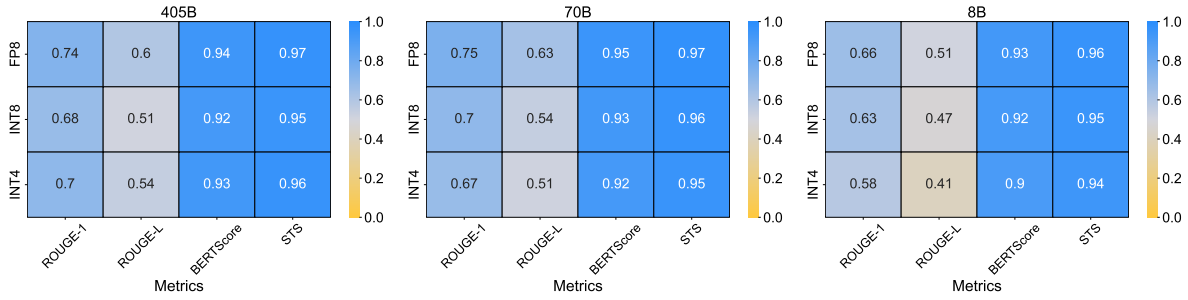

Figure 1: Text similarity metrics comparing the outputs of quantized Llama-3.1-Instruct models to full-precision baselines. We refer to W8A8-FP as FP8, W8A8-INT as INT8, and W4A16-INT as INT4.

(measuring chat and instruction-following performance, averaging two runs per model and quantization scheme), HumanEval, and HumanEval+ (measuring code generation quality and reporting pass@1 scores using the EvalPlus library (Liu et al., 2023a)), and RULER (evaluating long-context abilities). Table 3 summarizes the results.

On Arena-Hard-Auto-v0.1, quantized models exhibit competitive response quality, with overlapping 95% confidence intervals across all configurations (Appendix Table 7). In coding evaluations, quantized models also maintain strong performance, with 8-bit achieving 99.9% recovery and 4-bit recovering 98.9%, demonstrating their robustness across simple and complex coding tasks. Similarly, for the long-context RULER benchmark, quantized models achieve average score recovery of $\geq$ 98% across all formats. See Appendix A.1 for additional results.

4.3 Reasoning Benchmarks

Given the recent rise in popularity of reasoning abilities of LLMs, we also focus on the popular DeepSeek-R1-Distill (DeepSeek-AI, 2025) models. These models have been fine-tuned through the process of distillation for improved reasoning capabilities. To assess their reasoning performance, we focus on the challenging and widely recognized reasoning benchmarksi through LightEval (Habib et al., 2023): AIME 2024, MATH-500 (Lightman et al., 2023), and GPQA-Diamond (Rein et al., 2024). Following DeepSeek's recommendations for text generation, we use sampling with a temperature of 0.6 and top-p of 0.95, generating 20 responses per query to estimate the pass@1 score. The repetitive sampling was important to estimate an accurate average performance for the benchmarks due to high variance across the relatively small datasets. As can be seen from the results in Table 14, the conclusions from the previous sections with academic and real-world benchmarks still hold: **when quantization is properly tuned and configured, quantized models perform very competitively with their unquantized (BF16) baselines, recovering on average >99% accuracy except for the smallest models at INT4 which exhibit a bit larger but reasonable drops**.

4.4 Text Similarity Investigation

Next, we analyze the similarity of generated text between quantized and full-precision models. Using Arena-Hard-Auto-v0.1 prompts and greedy sampling for full reproducibility, we compute ROUGE-1, ROUGE-L, BERTScore, and Semantic Textual Similarity (STS) normalized to a 0-1 range.

As shown in Figure 1, large quantized models (70B and 405B) closely match their full-precision counterparts, achieving an average ROUGE-1 of 0.7 and ROUGE-L of 0.56, indicating strong word and structural preservation. BERTScore (0.93) and STS (0.96) further confirm semantic consistency despite minor token variations. While 8B models exhibit slightly higher variability, with ROUGE-1 and ROUGE-L dropping to 0.62 and 0.46, they still maintain strong semantic fidelity, as reflected in their BERTScore (0.92) and STS (0.95). **These results demonstrate that quantized models generate high-quality outputs across all sizes and schemes.**

5 Quantized Inference Performance

LLM inference consists of two main stages: prefill, where all input tokens are processed simultaneously, and decode, where tokens are generated sequentially. Prefill is typically compute-bound, while decode is memory-bound. Weight quantization primarily accelerates decode by reducing memory movement, whereas weight-and-activation quantization improves computational efficiency in prefill. Thus, the optimal choice for quantization scheme depends on the ratio of prefill to decode

Table 4: Detailed per-task and per-model breakdown of accuracy on the popular reasoning benchmarks across all quantized variants of DeepSeek-R1-Distill models from both Llama and Qwen families.

DeepSeek-R1-Distill		Recovery %	Average Score	AIME24 pass@1	MATH-500 pass@1	GPQA-Diamond pass@1
Llama-8B	BF16	100.0	62.9	49.3 ± 6.4	90.2 ± 1.2	49.3 ± 3.1
	W8A8-FP	100.6	63.3	50.8 ± 9.0	90.2 ± 1.1	48.7 ± 2.5
	W8A8-INT	99.6	62.7	49.1 ± 6.2	90.0 ± 1.0	48.9 ± 2.0
	W4A16-INT	97.2	61.1	46.3 ± 6.9	89.9 ± 1.1	47.1 ± 2.6
Llama-70B	BF16	100.0	76.2	67.8 ± 7.2	95.3 ± 0.7	65.6 ± 2.3
	W8A8-FP	100.3	76.5	69.2 ± 6.5	95.1 ± 0.5	65.2 ± 2.4
	W8A8-INT	99.7	76.0	67.8 ± 6.4	95.3 ± 0.5	65.0 ± 1.8
	W4A16-INT	98.3	75.0	65.6 ± 5.3	95.2 ± 0.6	64.0 ± 2.8
Qwen-32B	BF16	100.0	76.3	69.8 ± 4.9	95.1 ± 0.6	64.1 ± 2.1
	W8A8-FP	99.0	75.6	68.5 ± 4.0	95.3 ± 0.7	62.9 ± 2.6
	W8A8-INT	99.6	76.0	68.2 ± 5.1	95.0 ± 0.8	64.8 ± 2.6
	W4A16-INT	99.5	75.9	68.8 ± 4.2	95.0 ± 0.5	63.8 ± 1.7
Qwen-14B	BF16	100.0	73.6	66.7 ± 5.1	94.7 ± 0.7	59.4 ± 2.3
	W8A8-FP	101.0	74.3	68.1 ± 5.8	94.6 ± 0.6	60.1 ± 2.9
	W8A8-INT	99.4	73.1	66.3 ± 7.1	94.7 ± 0.7	58.3 ± 2.0
	W4A16-INT	99.0	72.8	66.0 ± 6.3	95.0 ± 0.5	57.5 ± 2.1
Qwen-7B	BF16	100.0	65.8	53.2 ± 6.4	93.7 ± 0.8	50.5 ± 2.8
	W8A8-FP	99.9	65.7	53.2 ± 7.5	93.6 ± 0.7	50.3 ± 2.0
	W8A8-INT	100.7	66.3	55.2 ± 4.9	93.0 ± 1.1	50.7 ± 3.5
	W4A16-INT	98.3	64.7	50.9 ± 7.8	93.3 ± 1.1	49.8 ± 2.8
Qwen-1.5B	BF16	100.0	50.0	30.1 ± 5.3	84.7 ± 1.1	35.4 ± 3.0
	W8A8-FP	100.3	50.2	29.8 ± 5.6	84.7 ± 1.3	35.9 ± 3.3
	W8A8-INT	96.9	48.5	26.7 ± 6.3	84.4 ± 1.1	34.4 ± 2.8
	W4A16-INT	93.5	46.8	24.6 ± 5.1	82.5 ± 1.1	33.2 ± 3.4

tokens. Beyond direct speedups, quantization also enhances end-to-end performance by increasing the number of simultaneous queries, improving efficiency, and enabling lower-cost GPU usage for memory-constrained tasks. Thus, real-world deployment involves complex trade-offs.

To assess these trade-offs, we benchmarked W8A8-FP, W8A8-INT, and W4A16-INT across three GPU types (A6000, A100, H100) in seven use cases. Tasks like code completion and instruction following involve short prefill phases (256 tokens) and varying decode lengths (1024 and 128 tokens, respectively). More complex tasks like summarization require significantly longer prefill (4096 tokens) with a moderate decode length (512 tokens). Multi-turn chat and RAG involve moderate prefill lengths (512 and 1024 tokens) with shorter decode phases (256 and 128 tokens). Finally, docstring generation (768 prefill, 128 decode) and code fixing (1024 prefill, 1024 decode) reflect intermediate token requirements. For latency-sensitive applications, we compare both synchronous and asynchronous deployment under latency constraints, while throughput-driven cases are evaluated in asynchronous mode. To assess cost efficiency across hardware setups, we use Lambda Labs' on-demand GPU pricing (Lambda Labs, 2024), shown in Table 9, which is standard.

5.1 Synchronous Deployment

Latency-sensitive applications are sometimes deployed in synchronous mode, where a single query is processed at a time. This approach minimizes latency by avoiding resource contention, making inference largely decode-bound.

Table 5 compares inference performance across model sizes, GPU types, quantization schemes, and use cases, highlighting the most cost-effective

Table 5: Synchronous inference performance comparison across model sizes and GPU configurations. Results show latency (in seconds) and cost-efficiency (Queries per USD) for various tasks. We refer to W8A8-FP as FP8, W8A8-INT as INT8, and W4A16-INT as INT4.

Size	GPU	#	Format	CR	Code Completion		Docstring Generation		Code Fixing		RAG		Instruction Following		Multi-Turn Chat		Summarization	
					Lat.	Q/$	Lat.	Q/$	Lat.	Q/$	Lat.	Q/$	Lat.	Q/$	Lat.	Q/$	Lat.	Q/$
8B	A6000	1	BF16	–	24.5	183	3.2	1,395	25.0	180	3.3	1,374	3.1	1,445	6.2	723	13.4	335
		1	INT8	1.54	15.9	284	2.1	2,157	16.3	276	2.1	2,139	2.0	2,249	4.0	1,120	8.9	506
		1	INT4	**2.39**	9.7	**462**	1.4	**3,290**	10.1	**445**	1.4	**3,136**	1.3	**3,543**	2.5	**1,787**	6.1	**736**
70B	A6000	4	BF16	–	61.7	18	6.6	170	62.6	18	8.1	138	8.0	141	15.8	71	32.6	35
		2	INT8	1.94	63.4	35	7.1	317	63.8	35	8.4	267	8.0	280	16.2	139	34.0	66
		2	INT4	**2.96**	39.2	**57**	5.0	**453**	40.4	**56**	5.8	**390**	5.1	**440**	10.2	**221**	23.5	**96**
	A100	2	BF16	–	50.7	20	2.9	343	51.2	20	6.8	148	6.4	156	12.9	78	27.3	37
		1	INT8	1.81	54.3	37	4.0	500	54.8	37	7.2	279	6.9	291	13.8	146	29.3	69
		1	INT4	**2.67**	35.0	**57**	2.8	**718**	35.8	**56**	5.2	**390**	4.6	**439**	9.2	**220**	21.0	**96**
	H100	2	BF16	–	31.3	18	4.0	139	31.5	18	4.1	138	4.0	142	7.9	71	16.4	34
		1	FP8	1.84	32.8	33	4.3	256	33.1	33	4.3	254	4.2	262	8.3	132	17.4	63
		1	INT4	**2.11**	28.6	**38**	3.8	**289**	28.2	**39**	3.8	**287**	3.7	**299**	7.1	**153**	15.3	**72**
405B	A100	16	BF16	–	81.9	2	10.8	12	81.2	2	11.2	11	10.6	12	20.9	6	44.1	3
		8	INT8	3.27	50.1	5	6.6	38	50.5	5	6.8	37	6.4	39	12.8	20	26.9	9
		4	INT4	**6.38**	48.9	**10**	7.0	**71**	49.5	**10**	7.3	**68**	6.4	**79**	12.7	**39**	29.4	**17**
	H100	16	BF16	–	50.6	1	6.5	12	50.3	1	6.6	11	6.4	12	13.0	6	26.5	3
		8	FP8	3.17	31.7	5	4.2	36	31.9	5	4.2	36	4.1	37	8.0	19	16.7	9
		4	INT4	**5.15**	37.5	**8**	5.0	**58**	37.8	**8**	5.1	**57**	4.8	**60**	9.2	**32**	20.4	**14**

†**CR**: Cost Reduction factor compared to BF16 baseline. Higher is better.
Lat.: Latency in seconds (lower is better). Q/$: Queries per USD (higher is better).

Table 6: Asynchronous inference performance evaluation across model sizes and hardware configurations. Results show throughput (queries per second) and cost-efficiency (queries per USD) for various use cases. We refer to W8A8-FP as FP8, W8A8-INT as INT8, and W4A16-INT as INT4.

Size	HW	Format	Speedup	Code Compl.		Doc. Gen.		Code Fixing		RAG		Inst. Following		Multi-Turn Chat		Summarization	
				QPS	Q/$	QPS	Q/$	QPS	Q/$	QPS	Q/$	QPS	Q/$	QPS	Q/$	QPS	Q/$
8B	1×A6000	BF16	–	1.5	6.8k	5.6	25.1k	1.1	4.8k	4.4	19.9k	11.8	53.0k	5.3	24.0k	0.7	3.2k
		INT8	**1.38**	2.2	9.8k	**7.7**	**34.6k**	**1.4**	**6.4k**	**6.1**	**27.6k**	**16.5**	**74.5k**	**7.2**	**32.3k**	**1.0**	**4.4k**
		INT4	1.08	**2.2**	**9.8k**	5.3	24.0k	1.3	6.0k	4.1	18.6k	11.2	50.5k	5.4	24.3k	0.7	3.1k
70B	4×A6000	BF16	–	0.4	0.4k	1.4	1.6k	0.3	0.3k	1.4	1.6k	3.3	3.8k	1.5	1.7k	0.2	0.3k
		INT8	1.91	0.7	0.8k	**3.9**	**4.4k**	0.5	0.6k	**2.8**	**3.1k**	**6.9**	**7.7k**	2.2	2.5k	**0.3**	**0.4k**
		INT4	**1.92**	**1.2**	**1.4k**	2.7	3.1k	**0.7**	**0.8k**	1.9	2.1k	5.2	5.9k	**2.6**	**3.0k**	0.3	0.3k
	4×A100	BF16	–	1.4	0.7k	6.9	3.5k	1.0	0.5k	3.3	1.6k	8.7	4.4k	4.3	2.2k	0.7	0.4k
		INT8	**1.87**	**2.4**	**1.2k**	15.9	8.0k	**1.8**	**0.9k**	**6.1**	**3.1k**	**16.5**	**8.3k**	**8.0**	**4.0k**	**1.2**	**0.6k**
		INT4	1.64	2.3	1.2k	**22.8**	**11.5k**	1.4	0.7k	4.3	2.2k	11.9	6.0k	5.8	2.9k	0.8	0.4k
	4×H100	BF16	–	3.5	1.0k	10.0	2.9k	2.6	0.7k	8.0	2.3k	20.3	5.9k	9.9	2.9k	1.7	0.5k
		FP8	**1.77**	**6.9**	**2.0k**	**17.8**	**5.2k**	**4.0**	**1.2k**	**14.3**	**4.2k**	**38.3**	**11.1k**	**18.4**	**5.4k**	**2.6**	**0.8k**
		INT4	1.55	5.9	1.7k	16.4	4.8k	3.1	0.9k	13.0	3.8k	35.8	10.4k	16.1	4.7k	2.2	0.6k
405B	16×A100	BF16	–	0.8	59	2.5	187	0.3	20	2.1	156	4.6	347	2.1	158	0.3	22
		INT8	**2.53**	1.3	98	**4.8**	**358**	1.1	79	**3.8**	**282**	**10.1**	**760**	**4.9**	**366**	**0.8**	**63**
		INT4	2.21	**1.9**	**144**	3.6	271	**1.2**	**93**	2.8	211	8.2	616	4.0	304	0.6	43
	16×H100	BF16	–	0.7	52	6.1	456	0.6	44	4.8	363	8.5	638	5.3	398	0.6	46
		FP8	3.04	**4.4**	**329**	9.6	725	**2.7**	**200**	7.6	571	20.7	1561	10.4	780	**1.7**	**125**
		INT4	**3.09**	4.0	304	**11.1**	**833**	2.5	192	**8.7**	**652**	**24.7**	**1856**	**11.6**	**872**	1.6	122

QPS: Queries per second (higher is better). Q/$: Queries per USD (higher is better).
Numbers denoted with *k* represent thousands (e.g., 20.3k = 20,300).

GPU configurations. The results show that W4A16-INT consistently achieves the highest performance gains across all models and hardware setups.

For 8B and 70B models, W4A16-INT reduces cost per query by 2–3× and improves latency by 1.5–2.5× compared to the full-precision BF16 baseline. The impact is even more pronounced at 405B, where W4A16-INT achieves 5–7× cost reductions and enables inference with fewer GPUs. Notably, deploying the 405B model on 4× A100 or H100

GPUs with W4A16-INT meets performance thresholds that previously required 16 GPUs in BF16, reducing inter-GPU communication and latency. Given the minor accuracy trade-offs observed in the previous section, this makes W4A16-INT highly effective for synchronous deployment.

5.2 Asynchronous Deployment

Processing multiple queries concurrently improves computational efficiency compared to single-query execution. vLLM automatically manages asynchronous requests, balancing computation between prefill and decode stages.

While asynchronous deployment increases per-query latency relative to synchronous execution, it amortizes computation across multiple requests, significantly boosting overall throughput, measured in queries per second (QPS). Table 6 reports the maximum achievable throughput and cost efficiency (queries per dollar) across different quantization formats, model sizes, and hardware configurations. The setups were optimized for peak BF16 performance and kept consistent when evaluating quantized models. **Results show that W8A8-INT and W8A8-FP yield the highest throughput, though W4A16-INT remains competitive and can outperform W8A8 in some scenarios.**

Many real-world applications impose latency constraints on asynchronous deployment. Figures 2 and 3 illustrate trade-offs between latency and throughput for two example tasks: docstring generation and code fixing. **W4A16-INT is more efficient at lower latencies, making it ideal for applications requiring rapid response times. In contrast, W8A8 formats maximize throughput at the cost of higher latency, making them better suited for batch processing**. The point where W8A8 overtakes W4A16 depends on factors such as model size, hardware, and task requirements.

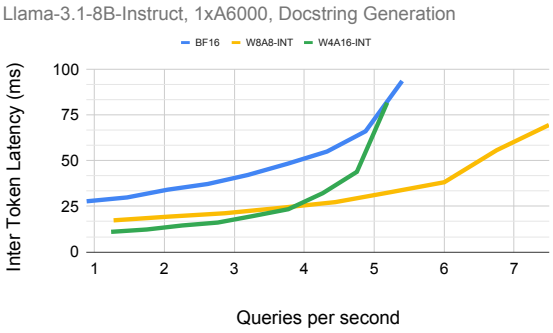

Figure 2: Latency-throughput example for docstring generation use-case. W4A16 is more efficient at low latency (lower throughput), whereas W8A8 becomes more efficient at high latency (high throughput).

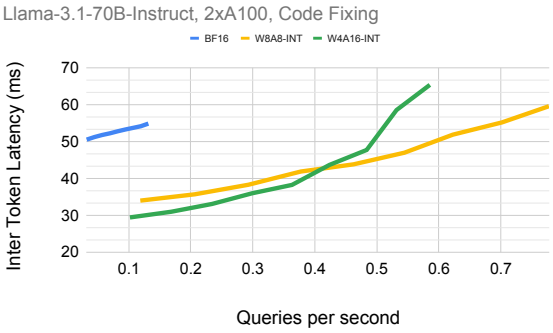

Figure 3: Latency-throughput example for code fixing use-case. W4A16 is more efficient at low latency (lower throughput), whereas W8A8 becomes more efficient at high latency (high throughput).

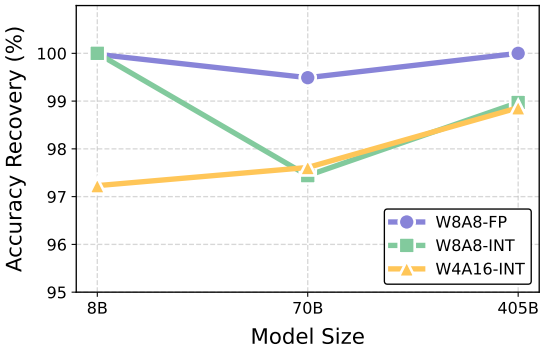

Figure 4: Accuracy recovery trends across academic benchmarks highlight the challenges of integer activation quantization, particularly at larger model sizes.

6 Conclusion

We provided a broad, in-depth study of accuracy-vs-performance-vs-cost trade-offs for quantized LLMs across various deployment environments, covering all quantization formats with efficient support, and a range of quantization algorithms, deployment use cases, and GPUs. In Figure 4 we summarize our findings in terms of accuracy recovery per quantization format, using carefully-tuned state-of-the-art quantization techniques. Broadly, our findings show that, with a judicious choice of algorithm and parametrization, these formats can offer higher accuracy than previously thought, significantly improve inference performance, and reduce costs. At the same time, we have also shown that the optimal choice of format can be task and algorithm specific, providing guidelines for this choice.

Limitations

While our study provides a comprehensive evaluation of quantization effects on model accuracy and inference performance, several limitations remain. We have primarily focused on weight and activation quantization, leaving open questions about the impact of compressing other model components such as the KV-cache, input embeddings, and language modeling head. Further investigation is needed to assess how these additional compression techniques influence both accuracy and computational efficiency. Additionally, our analysis does not fully explore the effects of quantization across specialized use cases, such as multi-lingual tasks, where accuracy degradation could vary significantly depending on the language distribution and underlying model architecture. Future work should extend these evaluations to provide a more holistic understanding of quantization trade-offs in diverse deployment scenarios.

References

Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. 2023. Towards end-to-end 4-bit inference on generative large language models. *arXiv preprint arXiv:2310.09259*.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. 2024. Quarot: Outlier-free 4-bit inference in rotated llms. *Preprint*, arXiv:2404.00456.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard (2023-2024). `https://huggingface.co/spaces/open-llm-leaderboard-old/open_llm_leaderboard`.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. 2023. Quip: 2-bit quantization of large language models with guarantees. *Preprint*, arXiv:2307.13304.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *Preprint*, arXiv:2403.04132.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. LLM.int8(): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. 2023. SpQR: A sparse-quantized representation for near-lossless llm weight compression. *arXiv preprint arXiv:2306.03078*.

Tim Dettmers and Luke Zettlemoyer. 2022. The case for 4-bit precision: k-bit inference scaling laws. *arXiv preprint arXiv:2212.09720*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. 2024. Extreme compression of large language models via additive quantization. *arXiv preprint arXiv:2401.06118*.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. 2024. Open llm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

Ruihao Gong, Yang Yong, Shiqiao Gu, Yushi Huang, Chentao Lv, Yunchen Zhang, Xianglong Liu, and Dacheng Tao. 2024a. Llmc: Benchmarking large language model quantization with a versatile compression toolkit. *Preprint*, arXiv:2405.06001.

Zhuocheng Gong, Jiahao Liu, Jingang Wang, Xunliang Cai, Dongyan Zhao, and Rui Yan. 2024b. What makes quantization for large language model hard? an empirical study from the lens of perturbation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18082–18089.

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. 2023. Lighteval: A lightweight framework for llm evaluation.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *Preprint*, arXiv:2103.03874.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.

Wei Huang, Xudong Ma, Haotong Qin, Xingyu Zheng, Chengtao Lv, Hong Chen, Jie Luo, Xiaojuan Qi, Xianglong Liu, and Michele Magno. 2024. How good are low-bit quantized llama3 models? an empirical study. *Preprint*, arXiv:2404.14047.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W Mahoney, and Kurt Keutzer. 2023. Squeezellm: Dense-and-sparse quantization. *arXiv preprint arXiv:2306.07629*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Lambda Labs. 2024. Lambda labs gpu cloud. Accessed: 2024-10-28.

Ariel N. Lee, Cole J. Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. 2024a. Owq: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. *Preprint*, arXiv:2306.02272.

Jemin Lee, Sihyeong Park, Jinse Kwon, Jihun Oh, and Yongin Kwon. 2024b. A comprehensive evaluation of quantized instruction-tuned large language models: An experimental analysis up to 405b. *arXiv preprint arXiv:2409.11055*.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. 2024a. Evaluating quantized large language models. *arXiv preprint arXiv:2402.18158*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024b. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2024c. From live data to high-quality benchmarks: The arena-hard pipeline.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024a. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. 2024b. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532*.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023a. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Peiyu Liu, Zikang Liu, Ze-Feng Gao, Dawei Gao, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2023b. Do emergent abilities exist in quantized large language models: An empirical study. *arXiv preprint arXiv:2307.08072*.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Compact language models via pruning and knowledge distillation. *arXiv preprint arXiv:2407.14679*.

Gunho Park, Baeseong Park, Se Jung Kwon, Byeongwook Kim, Youngjoo Lee, and Dongsoo Lee. 2022. nuQmm: Quantized matmul for efficient inference of large-scale generative language models. *arXiv preprint arXiv:2206.09557*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *Preprint*, arXiv:2311.12022.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. 2024. Musr: Testing the limits of chain-of-thought with multistep soft reasoning. *Preprint*, arXiv:2310.16049.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *Preprint*, arXiv:2210.09261.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. 2024a. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks. *Preprint*, arXiv:2402.04396.

Albert Tseng, Qingyao Sun, David Hou, and Christopher De Sa. 2024b. Qtip: Quantization with trellises and incoherence processing. *arXiv preprint arXiv:2406.11235*.

Mart van Baalen, Andrey Kuzmin, Markus Nagel, Peter Couperus, Cedric Bastoul, Eric Mahurin, Tijmen Blankevoort, and Paul Whatmough. 2024. Gptvq: The blessing of dimensionality for llm quantization. *arXiv preprint arXiv:2402.15319*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Preprint*, arXiv:2406.01574.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Julien Demouth, and Song Han. 2022. Smoothquant: Accurate and efficient post-training quantization for large language models. *arXiv preprint arXiv:2211.10438*.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *arXiv preprint arXiv:2206.01861*.

Zhewei Yao, Xiaoxia Wu, Cheng Li, Stephen Youn, and Yuxiong He. 2023. Zeroquant-v2: Exploring post-training quantization in llms from comprehensive study to low rank compensation. *arXiv preprint arXiv:2303.08302*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Ying Zhang, Peng Zhang, Mincong Huang, Jingyang Xiang, Yujie Wang, Chao Wang, Yineng Zhang, Lei Yu, Chuan Liu, and Wei Lin. 2024. Qqq: Quality quattuor-bit quantization for large language models. *arXiv preprint arXiv:2406.09904*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

A Additional Results

A.1 Real-World Benchmarks

In Figures 5 and 6 we report pass@10 scores for all models on HumanEval and HumanEval+ benchmarks.

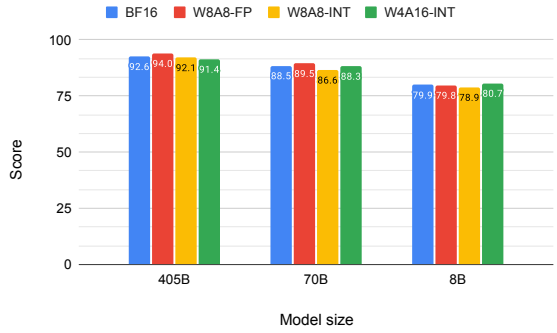

Figure 5: HumanEval pass@10 scores for quantized Llama-3.1-Instruct models.

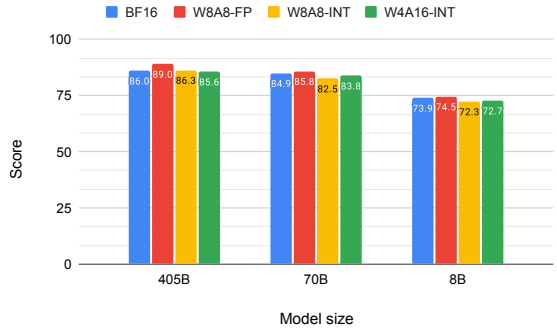

Figure 6: HumanEval+ pass@10 scores for quantized Llama-3.1-Instruct models.

In Table 7 we report scores of two Arena-Hard-Auto-v0.1 runs, aggregated average scores, and 95% confidence intervals (CI).

A.2 Detailed Comparison of GPTQ and AWQ

To complement the results in Table 1, Tables 8, 12, 13 provide a detailed per-task and per-run breakdown of scores.

A.3 GPU Pricing

We use Lambda Labs' on-demand GPU pricing (Lambda Labs, 2024), as displayed in Table 9. For A100 GPUs Lambda Labs only provides the 8x configuration. For scenarions with a smaller number of A100 GPUs we assume a price proportional to the number of GPUs.

Table 7: Scores and confidence intervals of two evaluation runs for Llama-3.1-Instruct models through Arena-Hard-Auto-v0.1.

Llama-3.1 Instruct	Score (1st run)	Score (2nd run)	Average Score	95% CI
BF16 405B	67.3	67.5	67.4	(-2.6, 1.9)
W8A8-FP	66.3	67.55	66.9	(-2.6, 2.3)
W8A8-INT	64.3	64.8	64.6	(-2.4, 2.8)
W4A16-INT	66.5	66.4	66.5	(-2.6, 2.3)
BF16 70B	55.8	58.2	57.0	(-2.6, 2.1)
W8A8-FP	57.6	57.75	57.7	(-2.4, 3.1)
W4A16-INT	57.1	56.8	57.0	(-2.8, 2.5)
W8A8-INT	56.0	56.6	56.3	(-2.9, 2.4)
BF16 8B	25.1	26.5	25.8	(-2.1, 2.1)
W8A8-FP	26.8	26.85	26.8	(-2.1, 2.6)
W8A8-INT	27.6	26.7	27.2	(-2.0, 2.2)
W4A16-INT	23.4	24.6	24.0	(-2.2, 2.0)

Table 8: Comparison of GPTQ and AWQ quantization algorithms, both with group size of 128, across two runs of the Arena-Hard-Auto-v0.1 benchmark.

	Score (1st run)	Score (2nd run)	Average Score
Llama-3.1-70B-Instruct	55.8	58.2	57.0
GPTQ (Frantar et al., 2022)	57.1	56.8	57.0
AWQ (Lin et al., 2024a)	56.3	57.0	56.3
Llama-3.1-8B-Instruct	25.1	26.5	25.8
GPTQ (Frantar et al., 2022)	23.4	24.6	24.0
AWQ (Lin et al., 2024a)	22.4	22.2	22.3

Table 9: On-demand hardware cost on Lambda Labs' cloud.

Hardware	On-demand cost (USD per hours)
1xA6000	0.80
2xA6000	1.60
4xA6000	3.20
8xA100	14.32
1xH100	3.29
2xH100	6.38
4xH100	12.36
8xH100	23.92

A.4 Academic Benchmarks

In Tables 10 and 11 we report accuracy recoveries per-task across academic benchmarks.

Table 10: Accuracy recoveries in percentages (%) for each task in the Open LLM Leaderboard V1 benchmark.

Llama-3.1-Instruct	MMLU 5-shot	MMLU CoT 0-shot	ARC-C 0-shot	GSM8k CoT 8-shot	HellaSwag 10-shot	Winogrande 5-shot	TruthfulQA 0-shot
Baseline BF16 8B	100.00	100.00	100.00	100.00	100.00	100.00	100.00
W8A8-FP	99.59	98.35	99.75	99.03	99.38	99.49	99.63
W8A8-INT	99.27	99.18	100.37	102.42	99.75	100.51	100.37
W4A16-INT	97.95	97.66	98.53	100.12	99.25	99.87	96.88
Baseline BF16 70B	100.00	100.00	100.00	100.00	100.00	100.00	100.00
W8A8-FP	100.00	99.42	100.21	99.58	99.77	99.18	99.84
W8A8-INT	99.88	99.77	99.79	99.26	99.88	99.77	101.15
W4A16-INT	99.76	99.53	99.46	99.47	99.42	100.23	98.52
Baseline BF16 405B	100.00	100.00	100.00	100.00	100.00	100.00	100.00
W8A8-FP	100.11	100.00	100.00	99.79	100.00	100.92	100.00
W8A8-INT	99.66	99.55	99.37	99.48	99.66	98.74	98.62
W4A16-INT	99.77	99.55	100.32	100.31	99.77	100.23	100.00

Table 11: Accuracy recoveries in percentages (%) for each task in the Open LLM Leaderboard V2 benchmark.

Llama-3.1-Instruct	IFEval 0-shot	BBH acc_norm 3-shot	Math lvl 5 exact_match 4-shot	GPQA acc_norm 0-shot	MuSR acc_norm 0-shot	MMLU-Pro acc 5-shot
Baseline BF16 8B	100.00	100.00	100.00	100.00	100.00	100.00
W8A8-FP	99.10	98.54	105.42	155.98	98.82	101.33
W8A8-INT	100.12	102.89	98.92	146.20	100.00	100.26
W4A16-INT	98.00	96.08	94.39	109.78	83.18	93.63
Baseline BF16 70B	100.00	100.00	100.00	100.00	100.00	100.00
W8A8-FP	101.34	98.44	107.52	94.68	94.49	99.13
W8A8-INT	100.17	98.89	91.83	88.38	92.62	97.86
W4A16-INT	99.22	98.60	93.52	89.94	94.99	98.19
Baseline BF16 405B	100.00	100.00	100.00	100.00	100.00	100.00
W8A8-FP	99.00	100.12	99.69	97.38	106.93	99.43
W8A8-INT	99.20	99.57	91.94	104.51	98.77	97.81
W4A16-INT	100.39	100.73	96.53	89.85	99.54	99.35

Table 12: Comparison of GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2024a) quantization algorithms, both with group size of 128, across Open LLM Leaderboard V1 benchmarks (Beeching et al., 2023) with Meta's prompts (Dubey et al., 2024).

	Average Score	MMLU 5-shot	MMLU CoT 0-shot	ARC-C 0-shot	GSM8k CoT 8-shot	HellaSwag 10-shot	Winogrande 5-shot	TruthfulQA mc2 0-shot
Llama-3.1-8B-Instruct	74.06	68.30	72.80	81.40	82.80	80.50	78.10	54.50
GPTQ	73.11	66.90	71.10	80.20	82.90	79.90	78.00	52.80
AWQ	72.69	66.37	69.76	80.89	82.56	79.61	76.80	52.81
Llama-3.1-70B-Instruct	84.20	82.37	86.06	93.30	94.90	86.80	85.30	60.70
GPTQ	83.77	82.03	85.54	92.80	94.40	86.30	85.50	59.80
AWQ	83.96	82.15	85.64	93.00	94.47	86.44	85.79	60.23

Table 13: Comparison of GPTQ and AWQ quantization algorithms, both with group size of 128, across Open LLM Leaderboard V2 benchmarks (Fourrier et al., 2024).

	Average Score	IFEval 0-shot	BBH acc_norm 3-shot	Math lvl 5 exact_match 4-shot	GPQA acc_norm 0-shot	MuSR acc_norm 0-shot	MMLU-Pro acc 5-shot
Llama-3.1-8B-Instruct	27.62	77.86	30.09	15.68	3.68	7.61	30.77
GPTQ (Frantar et al., 2022)	26.53	76.30	28.91	14.80	4.04	6.33	28.81
AWQ (Lin et al., 2024a)	27.40	78.25	27.20	13.87	5.21	10.45	29.41
Llama-3.1-70B-Instruct	41.66	86.41	55.79	26.07	15.40	18.16	48.12
GPTQ (Frantar et al., 2022)	40.58	85.74	55.01	24.38	13.85	17.25	47.25
AWQ (Lin et al., 2024a)	41.09	86.60	55.24	25.14	13.68	18.81	47.06

Table 14: Detailed per-task and per-model breakdown of accuracy on the popular reasoning benchmarks across all quantized variants of DeepSeek-R1-Distill models from both Llama and Qwen families.

DeepSeek-R1-Distill		Recovery %	Average Score	AIME24 pass@1	MATH-500 pass@1	GPQA-Diamond pass@1
Llama-8B	BF16	100.0	62.9	49.3 ± 6.4	90.2 ± 1.2	49.3 ± 3.1
	W8A8-FP	100.6	63.3	50.8 ± 9.0	90.2 ± 1.1	48.7 ± 2.5
	W8A8-INT	99.6	62.7	49.1 ± 6.2	90.0 ± 1.0	48.9 ± 2.0
	W4A16-INT	97.2	61.1	46.3 ± 6.9	89.9 ± 1.1	47.1 ± 2.6
Llama-70B	BF16	100.0	76.2	67.8 ± 7.2	95.3 ± 0.7	65.6 ± 2.3
	W8A8-FP	100.3	76.5	69.2 ± 6.5	95.1 ± 0.5	65.2 ± 2.4
	W8A8-INT	99.7	76.0	67.8 ± 6.4	95.3 ± 0.5	65.0 ± 1.8
	W4A16-INT	98.3	75.0	65.6 ± 5.3	95.2 ± 0.6	64.0 ± 2.8
Qwen-32B	BF16	100.0	76.3	69.8 ± 4.9	95.1 ± 0.6	64.1 ± 2.1
	W8A8-FP	99.0	75.6	68.5 ± 4.0	95.3 ± 0.7	62.9 ± 2.6
	W8A8-INT	99.6	76.0	68.2 ± 5.1	95.0 ± 0.8	64.8 ± 2.6
	W4A16-INT	99.5	75.9	68.8 ± 4.2	95.0 ± 0.5	63.8 ± 1.7
Qwen-14B	BF16	100.0	73.6	66.7 ± 5.1	94.7 ± 0.7	59.4 ± 2.3
	W8A8-FP	101.0	74.3	68.1 ± 5.8	94.6 ± 0.6	60.1 ± 2.9
	W8A8-INT	99.4	73.1	66.3 ± 7.1	94.7 ± 0.7	58.3 ± 2.0
	W4A16-INT	99.0	72.8	66.0 ± 6.3	95.0 ± 0.5	57.5 ± 2.1
Qwen-7B	BF16	100.0	65.8	53.2 ± 6.4	93.7 ± 0.8	50.5 ± 2.8
	W8A8-FP	99.9	65.7	53.2 ± 7.5	93.6 ± 0.7	50.3 ± 2.0
	W8A8-INT	100.7	66.3	55.2 ± 4.9	93.0 ± 1.1	50.7 ± 3.5
	W4A16-INT	98.3	64.7	50.9 ± 7.8	93.3 ± 1.1	49.8 ± 2.8
Qwen-1.5B	BF16	100.0	50.0	30.1 ± 5.3	84.7 ± 1.1	35.4 ± 3.0
	W8A8-FP	100.3	50.2	29.8 ± 5.6	84.7 ± 1.3	35.9 ± 3.3
	W8A8-INT	96.9	48.5	26.7 ± 6.3	84.4 ± 1.1	34.4 ± 2.8
	W4A16-INT	93.5	46.8	24.6 ± 5.1	82.5 ± 1.1	33.2 ± 3.4