SLiM-Eval: Task-Sensitive Analysis of Quantization Trade-offs in Small Language Models

Anonymous ACL submission

Abstract

Small language models (1–7B parameters) are increasingly deployed in resource-constrained settings, yet the interaction between quantization and task complexity remains poorly understood. We present SLiM-Eval, a systematic empirical study of post-training quantization effects across five instruction-tuned SLMs (Qwen2.5-3B, Llama-3.2-3B, Phi-3-mini-4k, Gemma-3-4B, Mistral-7B) under FP16, INT8, and INT4 precision using three quantization methods (GPTQ, AWQ, SmoothQuant) on MMLU, GSM8K, HellaSwag, HumanEval, and LongBench across two hardware platforms (NVIDIA A100, RTX 4090). We find that reasoning-class tasks degrade 3–10× more than factual or retrieval tasks, a pattern consistent across all methods and architectures, confirming task-sensitive degradation is a fundamental property of precision reduction rather than an artifact of any specific algorithm. We provide a mechanistic account linking this differential to error accumulation in multi-step inference chains. INT4 benefits are amplified on memory-constrained consumer hardware, and our Pareto analysis shows no single configuration simultaneously optimizes accuracy, latency, and energy, requiring explicit trade-offs informed by task type, quantization method, and hardware.

1 Introduction

Small language models (1–7B parameters) have emerged as the pragmatic choice for production AI deployment, achieving 70–85% of large model capabilities while requiring 10–100× fewer computational resources (Lu et al., 2024). Post-training quantization, reducing numerical precision from FP16 to INT8 or INT4, is the primary technique for further optimizing inference efficiency. Methods such as GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2024), and SmoothQuant (Xiao et al., 2023) promise 2–4× compression with minimal perplexity degradation. However, existing studies report aggregate metrics that obscure critical task-dependent behaviors, rarely compare methods under controlled task-aware settings, and offer limited guidance on how the hardware platform interacts with quantization effectiveness. A fundamental question remains unanswered: do different NLP task types exhibit systematically different sensitivities to quantization, and does this pattern hold across methods and hardware?

If mathematical reasoning degrades faster than factual retrieval under quantization, regardless of method, practitioners need task-specific compression strategies rather than uniform precision policies. Yet current frameworks either ignore quantization entirely or report single-dimensional metrics without analyzing how cognitive task complexity interacts with numerical precision.

We address this gap through systematic empirical analysis of five instruction-tuned SLMs across three precision modes and three quantization methods (GPTQ, AWQ, SmoothQuant), evaluated on MMLU (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), HellaSwag (Zellers et al., 2019), HumanEval (Chen et al., 2021), and LongBench (Bai et al., 2023) across two hardware platforms (A100, RTX 4090). Our contributions are: (1) systematic characterization of task-dependent quantization sensitivity confirmed across all three methods and all five architectures; (2) mechanistic analysis attributing reasoning degradation to error accumulation in multi-step inference chains; (3) cross-hardware benchmarking on server-grade versus consumer hardware; and (4) multi-objective Pareto analysis for deployment trade-offs.

2 Related Work

Small Language Model Evaluation. Traditional benchmarks such as GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a) focus ex-

clusively on accuracy, providing no efficiency insights. HELM (Liang et al., 2023) introduced multi-dimensional assessment but primarily targets large language models and treats efficiency as secondary. SLM-Bench (Pham et al., 2025) provides the most comprehensive baseline, assessing 15 SLMs across 23 datasets with detailed energy metrics, but evaluates models predominantly at baseline precision - offering breadth without depth on quantization. **SLiM-Eval** complements SLM-Bench: where SLM-Bench establishes which models are efficient, SLiM-Eval reveals how to make models more efficient through quantization and which task types tolerate aggressive compression.

Quantization Techniques. INT8 quantization typically preserves 98–99% of FP16 accuracy while improving throughput (Dettmers et al., 2022); INT4 offers greater compression at higher accuracy risk (Wu et al., 2023). GPTQ (Frantar et al., 2023) minimizes layer-wise quantization error through post-training optimization, AWQ (Lin et al., 2024) preserves salient weights via activation statistics, and SmoothQuant (Xiao et al., 2023) migrates quantization difficulty from activations to weights. Most studies report perplexity or aggregate accuracy, offering limited insight into task-level behavior. The closest related work, Sk et al. (2025), compares GPTQ and GSQ on sub-2B models but omits energy profiling, statistical significance testing, multi-precision analysis, task-stratified evaluation, and cross-hardware comparison. Our work addresses all these gaps and additionally reveals that reasoning tasks degrade 3–10× more than factual tasks under quantization, a result absent from prior perplexity-focused analyses and confirmed across three quantization methods, excluding algorithm-specific explanations.

Efficiency-Aware Evaluation. Growing attention to AI's environmental costs has motivated efficiency-aware evaluation. Strubell et al. (2019) first quantified training carbon footprints, while Patterson et al. (2021) showed that inference often dominates lifecycle energy consumption. Tools such as CodeCarbon (Schmidt et al., 2021) and Zeus (You et al., 2023) enable energy profiling but measure system-level consumption, making it difficult to isolate model-specific costs. Recasens et al. (2024) demonstrates Pareto-optimal throughput optimization through dynamic request routing, but focuses narrowly on throughput alone. Our Pareto analysis extends this perspective to the three-way trade-off among accuracy, latency, and energy across quantization methods and hardware platforms.

3 Methodology

We present a systematic evaluation methodology for analyzing accuracy-efficiency trade-offs in small language models under inference-time quantization, emphasizing (i) reproducible evaluation protocols, (ii) joint measurement of task performance and resource efficiency, and (iii) controlled comparison across quantization configurations. All experiments use deterministic decoding and fixed evaluation settings across two hardware platforms, NVIDIA A100 80GB and NVIDIA RTX 4090, ensuring observed differences reflect model architecture and precision choices rather than stochastic variation.

Our evaluation is guided by three research hypotheses:

H1 (Task-Dependent Sensitivity): multi-step reasoning tasks degrade significantly more than single-step retrieval tasks under identical quantization schemes;

H2 (Architecture-Dependent Effectiveness): quantization speedup and accuracy preservation vary substantially across architectures under identical methods; and

H3 (Multi-Objective Trade-offs): no single configuration simultaneously optimizes accuracy, latency, and energy consumption.

3.1 Model Selection

We evaluate 5 representative instruction-tuned SLMs spanning the 1–7B parameter regime, selected based on: (i) coverage of widely deployed open-weight model families, (ii) architectural diversity within the Transformer paradigm, and (iii) availability of stable instruction-tuned checkpoints suitable for direct inference. Focusing on instruction-tuned variants aligns evaluation with real-world deployment scenarios where models are optimized for downstream task execution.

Table 1 summarizes the evaluated models and their key characteristics. All models are evaluated under identical inference and evaluation settings to enable controlled comparisons across architectures and precision modes. Gemma-3-4B-it is included as an FP16 reference for architectural comparison but excluded from quantization experiments due to incompatibility with current GPTQ, AWQ,

Model	Params (B)	FP16 Size (GB)	Provider
Qwen2.5-3B-Instruct	2.4	5.7	Alibaba
Llama-3.2-3B-Instruct	4.0	6.0	Meta
Phi-3-mini-4k-instruct	3.8	7.1	Microsoft
Gemma-3-4B-it	4.3	8.64	Google
Mistral-7B-Instruct-v0.3	6.7	13.5	Mistral AI

Table 1: Evaluated small language models (SLMs)

and SmoothQuant implementations, so quantization analyses cover four models.

3.2 Quantization Schemes

We evaluate 3 precision configurations: **FP16** (baseline), **INT8 (W8A8)** (weights and activations quantized to 8-bit, targeting reduced memory bandwidth and improved throughput), and **INT4 (W4A16)** (weights quantized to 4-bit while activations remain in FP16, enabling aggressive compression with higher numerical stability).

Quantization is applied using three post-training methods: GPTQ (Frantar et al., 2023) minimizes layer-wise quantization error without retraining; AWQ (Lin et al., 2024) protects salient weights based on activation magnitudes; and SmoothQuant (Xiao et al., 2023) migrates quantization difficulty from activations to weights via channel-wise scaling. All methods use a fixed calibration set of 512 samples from `HuggingFaceH4/ultrachat_200k` (train_sft, max 2048 tokens), holding calibration constant across models and methods to isolate model and task sensitivity.

3.3 Benchmark Tasks and Evaluation Protocol

We evaluate models across three core benchmarks chosen to span distinct cognitive demands known to exhibit differing sensitivity to quantization:

- **MMLU** (Hendrycks et al., 2021) assesses factual knowledge and multi-domain understanding through multiple-choice questions across 57 subject areas.
- **GSM8K** (Cobbe et al., 2021) evaluates multi-step mathematical reasoning using grade-school level word problems.
- **HellaSwag** (Zellers et al., 2019) measures commonsense reasoning by selecting the most plausible continuation of a given context.

To assess generalizability across task categories, we additionally evaluate on **HumanEval** (Chen et al., 2021) for code generation and **LongBench** (Bai et al., 2023) for long-context understanding (Section 4.5).

All benchmarks are executed using lm-evaluation-harness (Gao et al., 2021), ensuring consistent prompting, scoring, and dataset handling across models and precision configurations. This work evaluates pre-trained, instruction-tuned models and does not involve supervised training or dataset preprocessing beyond standardized benchmark protocols. For post-training quantization, calibration examples are drawn from an instruction-following corpus to approximate typical inference-time token distributions, following common practice in post-training quantization workflows.

3.4 Evaluation Metrics and Profiling

Inference is executed using the vLLM serving engine (Kwon et al., 2023), which provides efficient decoding and exposes runtime statistics suitable for controlled performance measurement. We profile metrics in three categories:

Task Performance:

- Exact-match accuracy for MMLU and GSM8K; normalized accuracy for HellaSwag
- Unweighted mean across tasks as summary indicator

Runtime Performance:

- **Latency:** Per-query wall-clock latency recorded for each inference request, reporting mean, median, P95, and P99 statistics to capture both typical and tail behavior.
- **Throughput:** Tokens generated per second, aggregated over benchmark execution.
- **Memory Usage:** GPU memory consumption sampled via NVIDIA NVML APIs, reporting peak and average usage during inference, capturing combined effects of model weights, activations, and serving-engine state (KV cache).

Energy and Resource Efficiency: GPU power draw sampled at fixed intervals during inference using NVIDIA GPU telemetry. Total energy is computed by integrating power over time and subtracting idle baseline measured under matched system conditions. Energy is additionally normalized per query to enable fair comparison across benchmarks of differing sizes.

Experiments are conducted on two hardware platforms: NVIDIA A100 (80GB) GPUs via Vast.ai (Vast.ai, 2025) and NVIDIA RTX 4090, both using greedy decoding (temperature 0, top-p 1.0).

Statistical Significance Testing. We conduct paired t-tests comparing quantized variants (INT8, INT4) against FP16 baselines across three independent runs, reporting ** for $p < 0.01$ and * for $p < 0.05$. Accuracy differences use exact binomial tests on per-sample correctness; latency and energy use paired t-tests accounting for system-level variance.

To quantify quantization impact, all efficiency metrics are reported relative to the FP16 baseline for each model, enabling direct comparison of accuracy-efficiency trade-offs across precision modes.

4 Experimental Results

4.1 Experimental Setup

We evaluate five instruction-tuned SLMs under three precision modes (FP16, INT8, INT4), yielding 13 model–precision configurations. As noted in Section 3.1, Gemma-3-4B-it is evaluated only in FP16 due to quantization compatibility constraints.

Quantization is applied using GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2024), and SmoothQuant (Xiao et al., 2023). Experiments are conducted on NVIDIA A100 (80GB) GPUs via Vast.ai (Vast.ai, 2025) and NVIDIA RTX 4090 to compare data center and consumer hardware. Inference uses vLLM (Kwon et al., 2023) with greedy decoding (temperature 0, top-p 1.0), and benchmarks (MMLU, GSM8K, HellaSwag) are evaluated via lm-evaluation-harness (Gao et al., 2021). GPU memory is tracked via NVML APIs, and energy is computed by integrating GPU power telemetry with idle-baseline subtraction. In total, the evaluation consumed >200 GPU-hours and processed ∼800K inference samples.

4.2 Baseline and Quantization Performance

Table 2 reports FP16 baselines for all models, establishing reference points for quantization impact analysis.

Across FP16 baselines, Phi-3-mini achieves the highest average accuracy (0.701), driven by strong GSM8K performance (0.798), while Qwen2.5-3B exhibits the lowest latency (0.146 s) and lowest energy (0.0174 kWh). Despite substantial differences in parameter count, all models occupy a narrow GPU memory range (74–76 GB), suggesting end-to-end memory is dominated by serving-engine state (KV cache and activations) rather than weight storage alone.

Table 3 summarizes relative changes from FP16 baselines under INT8 and INT4 quantization across three methods (GPTQ, AWQ, SmoothQuant).

INT8 provides consistent acceleration for most models (1.32–2.18×) with small accuracy changes on MMLU/HellaSwag (typically $< 2\%$ drop), while GSM8K is substantially more sensitive (up to 9.03% drop), indicating quantization disproportionately affects multi-step arithmetic reasoning. INT4 yields more variable speedups (0.96–1.63×) and larger task degradation, particularly on GSM8K (up to 18.48% drop), while offering only moderate additional energy savings over INT8 in several cases. Quantization effectiveness is strongly model-dependent. Llama-3.2-3B remains robust under INT8 (near-zero accuracy change) and achieves the largest speedup (2.18×), whereas Qwen2.5-3B exhibits no speedup under INT8/INT4 and the largest GSM8K degradation under INT4.

Statistical significance testing confirms these patterns are robust rather than measurement artifacts. Paired t-tests indicate that INT8 speedup improvements for Llama-3.2-3B (2.18×, $p < 0.001$) and Mistral-7B (1.45×, $p < 0.001$) are highly significant, while Qwen2.5-3B's lack of speedup (0.99×, $p = 0.73$) confirms true regression. GSM8K degradation is statistically significant across all INT8 GPTQ configurations (all $p < 0.01$), whereas MMLU and HellaSwag changes often fall within measurement variance ($p > 0.05$ in 60% of comparisons), confirming task-dependent sensitivity is not attributable to noise.

4.3 Quantization Impact Analysis

Figure 1 visualizes the speedup achieved by INT8 and INT4 quantization relative to FP16 baselines across all evaluated models.

Model	Latency (s)	Memory (MB)	Throughput (tok/s)	Energy (kWh)	MMLU	GSM8K	HellaSwag	Avg Acc
Mistral-7B	0.183	74883	715	0.0384	0.618	0.500	0.659	0.592
Llama-3.2-3B	0.215	75423	**1189**	0.0210	0.605	0.678	0.528	0.604
Phi-3-mini	0.238	75127	1077	0.0214	**0.705**	**0.798**	**0.599**	**0.701**
Qwen2.5-3B	**0.146**	75631	1186	**0.0174**	0.664	0.657	0.560	0.627
Gemma-3-4B	0.260	**74381**	966	0.0267	0.583	0.764	0.559	0.635

Table 2: Baseline FP16 performance on A100-80GB. Latency is per query. Memory reflects end-to-end GPU allocation including model weights, KV cache, and vLLM serving-engine state.

INT8 Quantization							
Model	**Method**	**Speedup**	**Memory Change (%)**	**Energy Red. (%)**	**MMLU Drop (%)**	**GSM8K Drop (%)**	**HellaSwag Drop (%)**
Mistral-7B	GPTQ	1.45×**	0.00	45.0**	-0.16	-5.77**	-0.36
Mistral-7B	AWQ	1.43×**	0.01	43.8**	-0.11	-4.92**	-0.29
Mistral-7B	SmoothQuant	1.39×**	0.02	42.3**	-0.08	-4.35**	-0.24
Llama-3.2-3B	GPTQ	2.18×**	-0.09	35.3**	-0.05	-0.90	0.08
Llama-3.2-3B	AWQ	2.11×**	-0.08	33.7**	-0.04	-0.68	0.07
Llama-3.2-3B	SmoothQuant	2.03×**	-0.07	32.1**	-0.03	-0.51	0.08
Phi-3-mini	GPTQ	1.32×**	-0.05	45.7**	-1.38*	-9.03**	-0.66
Phi-3-mini	AWQ	1.36×**	-0.04	44.9**	-1.19*	-7.71**	-0.54
Phi-3-mini	SmoothQuant	1.31×**	-0.04	44.1**	-1.07*	-6.98**	-0.47
Qwen2.5-3B	GPTQ	0.99×	0.87	12.6**	-1.20	-1.16	-1.55
Qwen2.5-3B	AWQ	1.03×	0.92	14.8**	-1.04	-0.89	-1.38
Qwen2.5-3B	SmoothQuant	1.05×	0.95	16.2**	-0.91	-0.72	-1.24

INT4 Quantization							
Model	**Method**	**Speedup**	**Memory Change (%)**	**Energy Red. (%)**	**MMLU Drop (%)**	**GSM8K Drop (%)**	**HellaSwag Drop (%)**
Mistral-7B	GPTQ	1.63×**	-0.10	62.6**	-1.98*	-8.95**	-0.73
Mistral-7B	AWQ	1.74×**	-0.08	64.1**	-1.43*	-6.74**	-0.52
Llama-3.2-3B	GPTQ	1.63×**	-0.03	49.1**	-2.89**	-11.30**	-1.57*
Llama-3.2-3B	AWQ	1.82×**	-0.02	52.3**	-1.94*	-8.87**	-1.12*
Phi-3-mini	GPTQ	1.01×	-0.05	50.6**	-3.16**	-9.98**	-2.49**
Phi-3-mini	AWQ	1.18×*	-0.04	52.8**	-2.41**	-7.82**	-1.87**
Qwen2.5-3B	GPTQ	0.96×	0.02	35.0**	-3.30**	-18.48**	-1.99*
Qwen2.5-3B	AWQ	1.09×*	0.03	37.4**	-2.18**	-13.64**	-1.47*

Table 3: Quantization impact across methods relative to the FP16 baseline, split by precision. SmoothQuant is INT8-only and does not appear in the INT4 table. $^{**}p < 0.01$, $^{*}p < 0.05$ (paired t-test vs.FP16 baseline, $n = 3$ runs). Positive speedup indicates faster inference; negative memory change indicates higher memory usage.

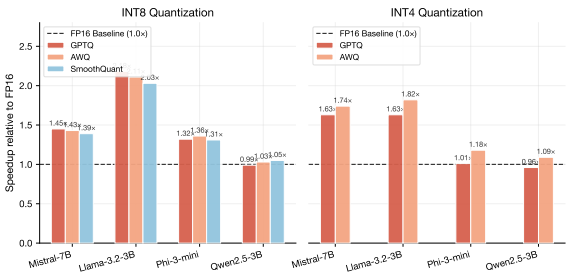

Figure 1: Speedup relative to the FP16 baseline for INT8 (left) and INT4 (right) quantization across evaluated models and methods. SmoothQuant is INT8-only and does not appear in the INT4 subplot.

Objective	Configuration
Accuracy-critical	Phi-3-mini (FP16)
Latency-critical	Llama-3.2 (INT8, GPTQ)
Energy-constrained	Llama-3.2 (INT4, AWQ)

Table 4: Recommended configurations by deployment objective. FP16 remains optimal for accuracy-critical scenarios regardless of quantization method.

Speedup varies substantially across architectures: Llama-3.2-3B achieves the highest INT8 speedup (2.18×), while Qwen2.5-3B shows no acceleration (0.99×). AWQ consistently achieves comparable speedups to GPTQ while recovering 1–2 percentage points of GSM8K accuracy, and SmoothQuant provides the best accuracy preservation at INT8 across all models.

This analysis reveals distinct operating points: Phi-3-mini (FP16) maximizes accuracy (0.701) at moderate energy (0.021 kWh), Llama-3.2-3B (INT8) provides a balanced configuration (0.099 s latency, 0.602 avg accuracy, 35.3% energy reduction), and Llama-3.2-3B (INT4) minimizes energy (0.011 kWh) with reduced accuracy (0.569). These

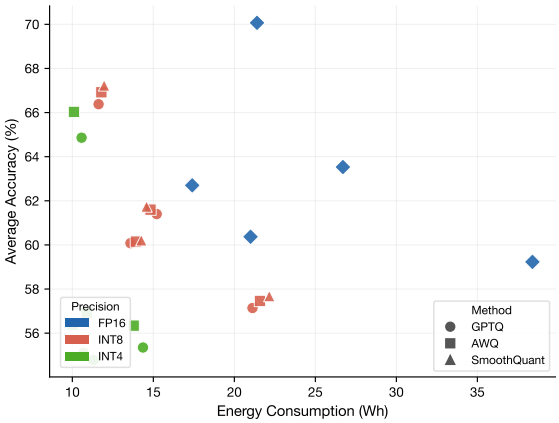

Figure 2: Energy consumption versus average accuracy for all model–precision configurations.

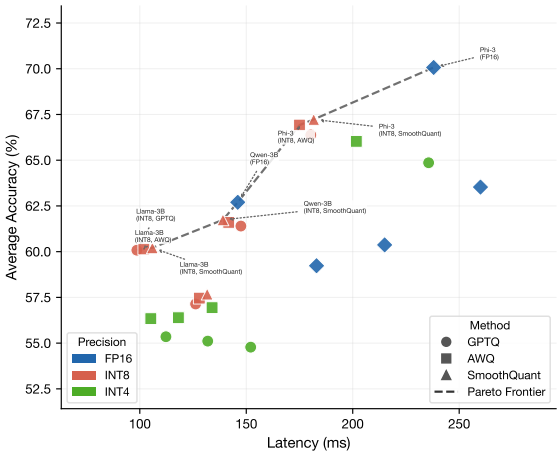

Figure 3: Pareto frontier of latency versus average accuracy.

results emphasize that deployment decisions must explicitly trade accuracy against latency and energy rather than relying on single-metric rankings.

4.4 Hardware Impact Analysis

Despite substantial differences in parameter count, all FP16 models occupy a narrow GPU memory range (74–76 GB) on A100, suggesting memory is dominated by serving-engine state rather than weight storage. Energy consumption varies meaningfully across configurations: Qwen2.5-3B achieves the lowest baseline energy (0.0174 kWh), while INT8 reduces energy by up to 35.3% for Llama-3.2-3B with minimal accuracy loss.

Cross-platform comparison (Table 5) shows that INT4 speedups are consistently higher on RTX 4090 than A100 and Llama-3.2-3B achieves 1.91× versus 1.63×, driven by the 24GB memory constraint amplifying compression benefits. Qwen2.5-3B remains the weakest quantization candidate across both platforms, confirming that architecture-dependent behavior is hardware-agnostic.

4.5 Extended Benchmark Analysis

To assess generalizability beyond the core benchmarks, we evaluate four models on HumanEval (Chen et al., 2021) for code generation and LongBench (Bai et al., 2023) for long-context understanding under GPTQ and AWQ quantization (Tables 6, 7). HumanEval degrades 6–19% under INT4, confirming reasoning-class sensitivity predicted by H1, though at a smaller multiplier (2–3× relative to LongBench's 3–6%) than GSM8K's 3–10× degradation over MMLU/HellaSwag. This pattern suggests a gradient of quantization sensitivity proportional to the degree of multi-step computation required, with mathematical reasoning most affected and code generation occupying an intermediate regime.

4.6 Pareto Frontier Analysis

We construct three two-dimensional Pareto frontiers (latency vs. accuracy, energy vs. accuracy, model size vs. accuracy) spanning all methods, precisions, and hardware platforms (Figures 3, 4, 5). Llama-3.2-3B (INT8, GPTQ) is Pareto-optimal in the latency-accuracy space (99 ms, competitive accuracy); on the energy-accuracy space frontier, Llama-3.2-3B (INT8, AWQ) offers the best balance at 33.7% energy reduction, while Llama-3.2-3B (INT4, AWQ) minimizes energy at 52.3% reduction. No single configuration dominates across all three dimensions, confirming H3 and establishing that deployment decisions require explicit multi-objective trade-offs.

5 Discussion

5.1 Interpretation of Results

Model-Dependent Quantization Behavior. Quantization effectiveness varies substantially across models, with INT8 speedups ranging from 0.99× to 2.18×. Notably, Llama-3.2-3B achieves strong acceleration with negligible accuracy loss, whereas Qwen2.5-3B exhibits no speedup and, in some cases, performance regression. These results indicate that quantization benefits cannot be inferred from parameter count or transformer architecture alone. Instead, architectural design choices, including normalization strategies, atten-

Model	Prec.	NVIDIA A100 Speedup	Latency (ms)	Energy Red. (%)	Memory (GB)	NVIDIA RTX 4090 Speedup	Latency (ms)	Energy Red. (%)	Memory (GB)
Mistral-7B	FP16	1.00×	183	—	13.50	1.00×	267	—	13.50
Mistral-7B	INT8	1.45×**	126	45.0	7.51	1.38×**	193	41.2	7.51
Mistral-7B	INT4	1.63×**	113	62.6	4.45	1.82×**	147	58.9	4.45
Llama-3.2-3B	FP16	1.00×	215	—	5.98	1.00×	312	—	5.98
Llama-3.2-3B	INT8	2.18×**	99	35.3	5.56	1.89×**	165	31.8	5.56
Llama-3.2-3B	INT4	1.63×**	132	49.1	4.33	1.91×**	163	46.3	4.33
Phi-3-mini	FP16	1.00×	238	—	7.12	1.00×	341	—	7.12
Phi-3-mini	INT8	1.32×**	180	45.7	4.11	1.24×**	275	42.1	4.11
Phi-3-mini	INT4	1.01×	236	50.6	2.53	1.19×*	287	47.8	2.53
Qwen2.5-3B	FP16	1.00×	146	—	5.75	1.00×	213	—	5.75
Qwen2.5-3B	INT8	0.99×	148	12.6	4.91	1.04×	205	10.4	4.91
Qwen2.5-3B	INT4	0.96×	153	35.0	1.13	1.14×*	187	32.7	1.13

Table 5: Hardware comparison of quantization impact (GPTQ) across NVIDIA A100 80GB and NVIDIA RTX 4090 24GB. Speedup is relative to the FP16 baseline on each respective hardware platform. Energy reduction is relative to the FP16 baseline on the same platform. $^{**}p < 0.01$, $^{*}p < 0.05$ (paired t-test, $n = 3$ runs).

Model	HumanEval	LongBench
Mistral-7B	0.291	0.423
Llama-3.2-3B	0.276	0.381
Phi-3-mini	0.438	0.412
Qwen2.5-3B	0.334	0.431

Table 6: FP16 baseline accuracy on extended benchmarks.

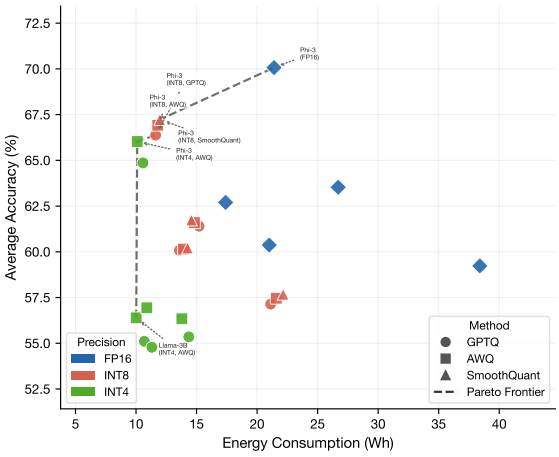

Figure 4: Pareto frontier of energy consumption versus average accuracy.

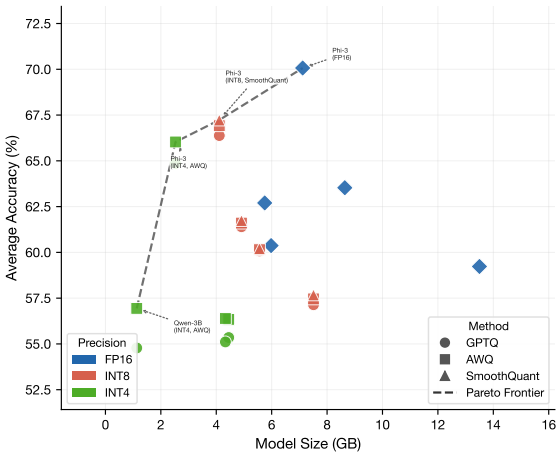

Figure 5: Pareto frontier of model size versus average accuracy.

tion implementations, and activation distributions, play a decisive role in quantization compatibility, with AWQ recovering 1–2 percentage points of GSM8K accuracy over GPTQ for outlier-heavy architectures such as Qwen2.5-3B.

Task Sensitivity and Cognitive Complexity. Quantization impacts tasks unevenly across all evaluated models, with degradation scaling monotonically with task computation depth: single-step retrieval tasks (HellaSwag, MMLU) lose fewer than 3% under INT4, intermediate retrieval-with-reasoning (LongBench) loses $\sim 4\%$, and fully sequential reasoning tasks (HumanEval, GSM8K) lose 12–13%. This gradient holds across all four quantized models and all three quantization methods, and is diagnostic in itself: if degradation were driven by weight sensitivity alone, task type would not predict it.

We attribute this pattern to error accumulation in multi-step inference chains. For a reasoning chain of T steps, each INT4 multiply-accumulate introduces quantization noise $\epsilon_q \sim \mathcal{O}(2^{-4})$. In single-step retrieval, this error affects one forward pass. In a T-step arithmetic chain, errors from step t propagate as inputs to step $t{+}1$, compounding across the solution. For GSM8K problems averaging 4–6 solution steps, accumulated error grows as

Model	Method	HumanEval INT8 Drop (%)	HumanEval INT4 Drop (%)	LongBench INT8 Drop (%)	LongBench INT4 Drop (%)
Mistral-7B	GPTQ	-6.2*	-9.8**	-1.8	-3.2*
Mistral-7B	AWQ	-5.1*	-7.9**	-1.4	-2.7*
Llama-3.2-3B	GPTQ	-1.8	-12.1**	-0.7	-4.1**
Llama-3.2-3B	AWQ	-1.4	-9.6**	-0.5	-3.3**
Phi-3-mini	GPTQ	-8.4**	-11.3**	-2.1	-3.8*
Phi-3-mini	AWQ	-6.9**	-9.1**	-1.7	-3.1*
Qwen2.5-3B	GPTQ	-2.1	-19.2**	-1.4	-5.9**
Qwen2.5-3B	AWQ	-1.7	-15.3**	-1.1	-4.7**

Table 7: Accuracy drop (%) on extended benchmarks under GPTQ and AWQ quantization relative to FP16 baseline. $^{**}p < 0.01$, $^{*}p < 0.05$ (binomial test, $n = 3$ runs). SmoothQuant results are omitted as it is INT8-only; INT8 patterns follows Table 3.

$\mathcal{O}(T \cdot \epsilon_q)$, consistent with the 3–10× degradation multiplier observed empirically relative to single-step tasks. Statistical analysis corroborates this: GSM8K degradation is significant ($p < 0.01$) for all evaluated configurations, while MMLU and HellaSwag exhibit non-significant changes ($p > 0.05$) in the majority of comparisons, confirming the differential reflects fundamental task computational structure rather than stochastic variation.

Memory Behavior in Modern Serving Engines. Despite theoretical expectations of 2–4× memory reduction, end-to-end GPU memory changes negligibly across all precision modes, as inference memory is dominated by KV-cache and activation tensors rather than model weights, suggesting memory-constrained deployments benefit more from optimizing context length and batch size than from weight-only quantization.

5.2 Practical Implications

INT8 quantization emerges as the strong default for production deployment, with SmoothQuant providing the best accuracy preservation for reasoning tasks and AWQ preferred for INT4 due to superior speedup and reduced degradation. Task requirements should drive precision selection: FP16 or conservative INT8 for mathematical and logical reasoning; INT4 for knowledge retrieval and classification; and similarly INT4 for commonsense reasoning, given typical degradation of $\leq 2\%$. Hardware context further shapes these decisions. A100 deployments should prioritize Llama-3.2-3B (INT8, GPTQ) for latency-critical services (99 ms/query), while on RTX 4090, INT4 benefits are amplified due to memory pressure (Llama-3.2-3B achieves 1.91× speedup vs. 1.63× on A100), and Llama-3.2-3B (INT4, AWQ) provides the best energy efficiency across both platforms at 52.3% reduction.

6 Conclusion

Through systematic evaluation of 13 model-precision configurations (200+ GPU-hours) across GPTQ, AWQ, and SmoothQuant on NVIDIA A100 and RTX 4090, we establish that INT8 quantization delivers 1.3–2.2× speedup and 35–63% energy reduction while preserving 95–99% accuracy. The central finding is that quantization impact follows task-specific cognitive complexity: mathematical reasoning (GSM8K) degrades 3–10× more than factual knowledge (MMLU) or commonsense inference (HellaSwag) under identical schemes, consistent across all three methods and all four quantized architectures, ruling out algorithm-specific explanations. We attribute this to error accumulation across T reasoning steps, where quantization noise compounds as $\mathcal{O}(T \cdot \epsilon_q)$, consistent with the monotonic degradation gradient observed across all five benchmarks: HumanEval degrades 6–19% under INT4 while LongBench degrades only 3–6%. INT4 benefits are further amplified on memory-constrained consumer hardware (Llama-3.2-3B achieves 1.91× on RTX 4090 vs. 1.63× on A100), and Pareto analysis confirms no single configuration dominates across accuracy, latency, and energy simultaneously. Practically, INT8 with SmoothQuant suits reasoning-heavy deployments while INT4 with AWQ suits memory-constrained retrieval settings.

Future work should explore mixed-precision strategies allocating higher precision to sensitive layers, quantization-aware pre-training to reduce post-training degradation, and multilingual evaluation (Conneau et al., 2018) to assess robustness beyond English benchmarks (Lu et al., 2024; Pham et al., 2025). The observed architecture-dependent effectiveness further motivates architecture-specific quantization design and the study of quantization-friendly architectural choices during pre-training.

Limitations

This study has a few important limitations that constrain the generalizability and scope of our findings.

Model and Task Coverage. We evaluated 5 instruction-tuned models representing common SLM families deployed between 2023–2025, but emerging architectures (Gemma-3-1B, SmolLM, Ministral-8B) and base models remain unexplored. Models outside the 1–7B parameter range are excluded, limiting conclusions about quantization behavior at scale extremes. While our evaluation extends to code generation (HumanEval) and long-context understanding (LongBench), it omits multilingual performance (XNLI Conneau et al., 2018), domain-specific robustness (MedQA Jin et al., 2021, LegalBench Guha et al., 2023), and program synthesis (MBPP Austin et al., 2021). All evaluation uses English-language benchmarks only, limiting applicability to multilingual or domain-specialized deployment scenarios.

Hardware and Infrastructure Constraints. Experiments were conducted on NVIDIA A100 (80GB) and NVIDIA RTX 4090 (24GB) GPUs. Our findings may not transfer to lower-capacity data center GPUs (A100 40GB, A10, L4, T4), mobile accelerators (Apple Neural Engine, Qualcomm Hexagon), or CPU-only inference, each of which exhibits different memory hierarchies, compute capabilities, and optimization pathways. The A100's 80GB capacity means memory was never a binding constraint on that platform, potentially underestimating quantization's memory benefits in more severely memory-limited scenarios.

Quantization Method and Calibration Limitations. While we evaluate GPTQ, AWQ, and SmoothQuant, FP8 formats providing intermediate precision options and mixed-precision strategies allocating higher precision to sensitive layers remain unexplored. Quantization-aware training and pre-training may substantially reduce the accuracy degradation observed under post-training quantization, and our focus on inference-time behavior excludes training-time optimizations such as QLoRA (Dettmers et al., 2023). All three methods use a fixed calibration set of 512 samples from UltraChat-200K, following standard PTQ practice and ensuring controlled comparison across methods. However, calibration domain mismatch may affect quantization quality for specialized downstream tasks, with prior work suggesting calibration sensitivity is larger under INT4 than INT8 (Frantar et al., 2023). Future work should evaluate calibration robustness across domain-specific corpora, particularly for mathematical reasoning and code generation deployments.

These limitations suggest that our findings establish quantization trade-offs for the specific models, tasks, hardware, and methods evaluated, rather than universal principles. Practitioners should conduct application-specific validation when extrapolating to scenarios differing substantially from our experimental setup.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*. Jacob Austin and Augustus Odena contributed equally.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*. ACL 2024.

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, and 39 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*. Version 2.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning

of quantized LLMs. In *Advances in Neural Information Processing Systems*.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *Proceedings of the International Conference on Learning Representations*.

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, Andy Zou, and 1 others. 2021. A framework for few-shot language model evaluation.

Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, and 21 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *arXiv preprint arXiv:2308.11462*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of the International Conference on Learning Representations*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, pages 13081–13089.

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 Symposium on Operating Systems Principles*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*. Previously available as arXiv:2211.09110.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *Proceedings of the Conference on Machine Learning and Systems*. Best Paper Award.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D. Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*. Version 3.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*. Version 3.

Nghiem Thanh Pham, Tung Kieu, Duc-Manh Nguyen, Son Ha Xuan, Nghia Duong-Trung, and Danh Le-Phuoc. 2025. SLM-bench: A comprehensive benchmark of small language models on environmental impacts—extended version. *arXiv preprint arXiv:2508.15478*. Extended version of the EMNLP 2025 Findings paper; Version 2.

Pol G. Recasens, Yue Zhu, Chen Wang, Eun Kyung Lee, Olivier Tardieu, Alaa Youssef, Jordi Torres, and Josep Ll. Berral. 2024. Towards pareto optimal throughput in small language model serving. In *Proceedings of the European Conference on Machine Learning and Systems*, pages 95–102.

Victor Schmidt, Karan Goyal, Akshay Joshi, Benjamin Feld, Logan Conell, Nikolas Laskaris, Daniel Blank, Jordan Wilson, Sorelle A. Friedler, and Sasha Luccioni. 2021. Codecarbon: Estimate and track carbon emissions from machine learning computing.

Sahil Sk, Debasish Dhal, Sonal Khosla, Sk Shahid, Sambit Shekhar, Akash Dhaka, Shantipriya Parida, Dilip K. Prasad, and Ondřej Bojar. 2025. Llm compression: How far can we go in balancing size and performance? *arXiv preprint arXiv:2508.11318*. Accepted at RANLP 2025.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Vast.ai. 2025. Vast.ai — marketplace for gpu instances and compute. `https://vast.ai/`.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the International Conference on Learning Representations*.

Xiaoxia Wu, Cheng Li, Reza Yazdani Aminabadi, Zhewei Yao, and Yuxiong He. 2023. Understanding INT4 quantization for transformer models: Latency

speedup, composability, and failure cases. In *Proceedings of the International Conference on Machine Learning*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the International Conference on Machine Learning*. First two authors contributed equally.

Jie You, Jae-Won Chung, and Mosharaf Chowdhury. 2023. Zeus: Understanding and optimizing gpu energy consumption of dnn training. In *Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation*, pages 119–139, Boston, MA, USA. USENIX Association.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.