SLiM-Eval: Evaluating Quantization Trade-offs in Small Language Models

Kavin Aravindhan Rajkumar
Department of Computer Science
Columbia University
New York, USA
kr3131@columbia.edu

Vishruth Devan
Department of Computer Science
Columbia University
New York, USA
vd2461@columbia.edu

***Abstract*—Small language models (SLMs, 1-7B parameters) are increasingly deployed in production systems where efficiency is as critical as accuracy, yet existing evaluations largely ignore deployment-critical metrics such as latency, throughput, memory, and energy. We present SLiM-Eval, a systematic evaluation of accuracy–efficiency trade-offs under quantization, benchmarking five instruction-tuned SLMs (Qwen2.5-3B, Llama-3.2-3B, Phi-3-mini-4k, Gemma-3-4B, and Mistral-7B) across FP16, INT8, and INT4 precision on MMLU, GSM8K, and HellaSwag, with over 200 hours of controlled experiments on NVIDIA A100 GPUs. Our results show that INT8 achieves 1.3–$2.2\times$ speedup with $< 2\%$ accuracy loss for most models, while mathematical reasoning degrades 3–$10\times$ more than factual tasks under quantization, and quantization effectiveness varies strongly by architecture ($2.18\times$ speedup for Llama-3.2-3B vs. $0.99\times$ regression for Qwen2.5-3B). Pareto frontier analysis reveals unavoidable trade-offs between accuracy, latency, and energy, identifying Llama-3.2-3B (INT8) as the best balanced configuration (99 ms latency, 60.5% accuracy, 49% energy reduction), Phi-3-mini (FP16) as accuracy-optimal (70.1%), and Llama-3.2-3B (INT4) as energy-optimal (0.011 kWh, 56.9%). We release SLiM-Eval as an open-source framework for reproducible, efficiency-aware SLM deployment evaluation.**

***Index Terms*—Small language models, quantization, efficiency evaluation, model compression, inference optimization, deployment optimization, Pareto analysis**

I. Introduction

A. Background and Motivation

Small language models (SLMs) with 1–7 billion parameters are increasingly adopted for production AI systems due to their favorable balance between capability and efficiency. Recent models such as Phi-3 [6], Llama-3.2 [7], Qwen-2.5, and Gemma-3 demonstrate that carefully designed compact architectures can achieve 70–85% of large language model (LLM) performance while requiring 10–100× fewer computational resources [1], [2]. As a result, SLMs are widely deployed in latency-sensitive, energy-constrained, and cost-critical settings, including on-device assistants, edge inference, and scalable production services where LLMs are impractical.

Despite this shift, existing evaluation frameworks remain misaligned with deployment realities. Benchmarks such as MMLU [22] and HellaSwag [24], as well as holistic frameworks like HELM [3], primarily emphasize accuracy while treating efficiency metrics like latency, throughput, memory usage, and energy consumption, as secondary or absent. This evaluation paradigm creates a critical blind spot: a model achieving 65% accuracy with 50ms latency and 2GB memory may vastly outperform a 70%-accurate alternative requiring 500ms and 8GB in production environments where user experience, throughput, and operational costs dominate.

The situation grows more complex when considering quantization – the primary technique for optimizing SLMs through reduced numerical precision (FP16 $\rightarrow$ INT8 $\rightarrow$ INT4). While quantization promises $2-4\times$ speedups and memory reductions, its impact remains poorly characterized across the accuracy-efficiency landscape. Practitioners face unanswered questions: Which models benefit most? How much accuracy degradation for given speedup? Do mathematical reasoning tasks degrade differently than factual recall? Without systematic answers, deployment decisions rely on trial-and-error.

B. Problem Statement

Current SLM evaluation practices suffer from three fundamental and interconnected gaps:

Gap 1: Efficiency-Blind Benchmarking. Standard evaluation protocols measure correctness (accuracy, F1, BLEU) while ignoring dimensions critical for deployment: inference latency, throughput, memory footprint, and energy consumption. A model ranking first on MMLU may rank last in production viability due to prohibitive latency or energy costs.

Gap 2: Absence of Quantization-Aware Evaluation. While quantization is ubiquitous in production systems, existing benchmarks evaluate models predominantly at baseline precision (FP16/FP32). The relationship between quantization aggressiveness and task-specific accuracy degradation remains uncharacterized, forcing practitioners into conservative precision choices that sacrifice efficiency gains.

Gap 3: Lack of Multi-Objective Optimization Frameworks. SLM deployment inherently requires balancing conflicting objectives like maximizing accuracy while minimizing latency, memory, and energy. However, benchmarks report single-dimensional rankings rather than Pareto frontiers revealing optimal trade-offs.

C. Objectives and Scope

To address these gaps, we introduce SLiM-Eval, a systematic framework for evaluating SLMs across accuracy–efficiency trade-offs under quantization. Our objectives are threefold:

- to establish reproducible evaluation protocols that jointly measure task accuracy and deployment-critical efficiency metrics
- to quantify the impact of FP16, INT8, and INT4 quantization on model performance across diverse task types
- to expose multi-objective trade-offs using Pareto frontier analysis to support informed deployment decisions

We evaluate five representative instruction-tuned SLMs - Qwen2.5-3B, Llama-3.2-3B, Phi-3-mini-4k, Gemma-3-4B, and Mistral-7B, across three NLP benchmarks (MMLU, GSM8K, and HellaSwag). Unlike prior work emphasizing broad model coverage [2], our study prioritizes depth, providing focused, quantization-aware insights into optimizing SLMs for real-world deployment scenarios.

II. Literature Review

A. *Review of Relevant Literature*

Small language models (SLMs) reflect a shift from scale-driven performance toward efficiency-aware design. Early approaches relied on distillation, with DistilBERT and TinyBERT achieving substantial parameter reductions while preserving most task accuracy [4], [5]. More recent models emphasize data quality and architectural choices: Microsoft's Phi series demonstrates strong reasoning performance at 1–4B parameters [6], while Meta's Llama 3.2 achieves competitive results through extended pretraining [7]. Despite these advances, standard benchmarks such as GLUE and SuperGLUE focus exclusively on task accuracy [8], [9]. HELM introduced multi-dimensional evaluation across accuracy and robustness, but primarily targets large language models and treats efficiency as a secondary concern without quantization-aware analysis [3]. SLM-Bench extends evaluation to small language models and environmental impact [2], but largely evaluates models at baseline precision and does not examine how optimization techniques such as quantization alter accuracy–efficiency trade-offs.

Quantization is a key technique for improving inference efficiency by reducing numerical precision. Post-training quantization methods are widely used due to their practicality, with INT8 commonly preserving near-FP16 accuracy while improving throughput [11]. More aggressive INT4 schemes offer greater compression but introduce higher risk of accuracy degradation [10]. GPTQ has emerged as a leading post-training method, enabling low-bit quantization with minimal perplexity loss [12]. Alternative approaches such as AWQ and SmoothQuant improve robustness by accounting for activation outliers [14], [15]. However, most quantization studies emphasize language modeling metrics or isolated benchmarks, offering limited insight into downstream task performance and deployment-level efficiency trade-offs. While QLoRA demonstrates the effectiveness of INT4 for fine-tuning [16], it primarily addresses training efficiency rather than inference-time behavior.

Growing awareness of the environmental and economic costs of AI has motivated efficiency-aware evaluation. Early work quantified the carbon footprint of training large models [17], while later studies showed that inference often dominates lifecycle energy consumption [18]. Tools such as CodeCarbon and Zeus enable energy measurement at the system level [19], [20], but do not isolate model-specific costs under controlled conditions. Recent systems work explores Pareto-optimal serving strategies, such as routing requests across quantized variants to improve throughput [21]. However, existing studies typically optimize single dimensions and do not jointly consider accuracy, latency, and energy trade-offs.

B. *Gaps in Existing Research*

Despite progress in SLM development, quantization, and efficiency measurement, three key gaps remain:

- **Lack of quantization-aware SLM evaluation:** Existing benchmarks largely evaluate models at baseline precision, leaving accuracy–efficiency trade-offs under INT8 and INT4 poorly characterized.
- **Limited task-sensitive analysis:** Prior work does not systematically examine how quantization affects different task types, such as factual knowledge versus multi-step reasoning.
- **Absence of multi-objective deployment frameworks:** Current evaluations report isolated metrics rather than Pareto-optimal trade-offs across accuracy, latency, and energy.

Our work addresses these gaps through systematic, quantization-aware evaluation and multi-objective analysis tailored to real-world SLM deployment.

III. Methodology

We present a systematic evaluation methodology for analyzing accuracy–efficiency trade-offs in SLMs under practical inference-time optimizations. The methodology is designed to support deployment-oriented analysis by emphasizing (i) reproducible evaluation protocols, (ii) joint measurement of task performance and resource efficiency, and (iii) controlled comparison across quantization configurations. Rather than introducing new training or architectural modifications, SLiM-Eval focuses exclusively on inference-time behavior, reflecting the dominant optimization pathway used in production SLM deployments.

The evaluation pipeline consists of four stages: (1) preparing model variants under defined precision schemes, (2) executing standardized benchmark evaluations, (3) profiling runtime performance and memory behavior, and (4) measuring energy consumption under controlled conditions. All experiments are conducted with deterministic decoding and fixed evaluation settings to ensure that observed differences arise from model architecture and precision choices rather than stochastic variation.

A. Data Collection and Preprocessing

This work evaluates pre-trained, instruction-tuned language models and does not involve supervised training or dataset preprocessing beyond standardized benchmark protocols. All task evaluations are performed using `lm-evaluation-harness`, which provides consistent prompting templates, few-shot selection strategies, and scoring logic across models and tasks. This approach ensures that comparisons across precision modes and architectures are not confounded by differences in data formatting or evaluation methodology.

For post-training quantization, a fixed calibration dataset is used without modifying benchmark inputs. Calibration examples are drawn from an instruction-following corpus to approximate typical inference-time token distributions, following common practice in post-training quantization workflows.

B. Model Selection

We evaluate five representative instruction-tuned SLMs spanning the 1–7B parameter regime. Models are selected based on three criteria: (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 the evaluation with real-world deployment scenarios, where models are typically optimized for downstream task execution rather than raw language modeling.

Table I summarizes the evaluated models and their key characteristics. All models are evaluated under identical inference and evaluation conditions to enable controlled, apples-to-apples comparisons across architectures and precision modes.

TABLE I
EVALUATED SMALL LANGUAGE MODELS

Model	Params (B)	FP16 Size (GB)	Provider
Qwen2.5-3B-Instruct	2.4	4.5	Alibaba
Llama-3.2-3B-Instruct	4.0	7.4	Meta
Phi-3-mini-4k-instruct	3.8	7.1	Microsoft
Gemma-3-4B-it	∼4.0*	∼8.0	Google
Mistral-7B-Instruct-v0.3	6.7	12.5	Mistral AI

*Parameter count not explicitly reported in the model configuration.

C. Optimization Procedures

We optimize inference performance only, reflecting the dominant optimization pathway for deployed SLMs. Each model is evaluated under three precision configurations representing commonly used deployment choices:

- **FP16 (Baseline):** Half-precision inference serves as the reference configuration for accuracy and efficiency comparisons.
- **INT8 (W8A8):** Post-training quantization is applied to both weights and activations using 8-bit integer precision, targeting reduced memory bandwidth and improved throughput while preserving task accuracy.
- **INT4 (W4A16):** Weights are quantized to 4-bit precision while activations remain in FP16, enabling more aggressive compression while maintaining higher numerical stability during forward computation.

Quantization is performed using GPTQ [12], a widely adopted post-training quantization method that minimizes layer-wise quantization error without retaining. GPTQ is implemented via `llmcompressor` using a fixed calibration set of 512 instruction samples from the `HuggingFaceH4/ultrachat_200k` dataset (`train_sft` split), with a maximum calibration sequence length of 2048 tokens. Holding calibration size and sequence length constant across models ensures that observed differences reflect model and task sensitivity rather than calibration variability.

D. Benchmark Tasks

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

- **MMLU** [22] assesses factual knowledge and multi-domain understanding through multiple-choice questions across 57 subject areas.
- **GSM8K** [23] evaluates multi-step mathematical reasoning using grade-school level word problems.
- **HellaSwag** [24] measures commonsense reasoning by selecting the most plausible continuation of a given context.

All benchmarks are executed using `lm-evaluation-harness` [25], ensuring consistent prompting, scoring, and dataset handling across models and precision configurations. Few-shot settings follow established benchmark conventions.

E. Profiling Tools and Measurement Protocols

Inference is executed using the `vLLM` serving engine [26], which provides efficient decoding and exposes runtime statistics suitable for controlled performance measurement. We profile three categories of deployment-relevant metrics:

- **Latency:** Per-query wall-clock latency is recorded for each inference request. To capture both typical and tail behavior, we report mean, median, P95, and P99 latency statistics.
- **Throughput:** Throughput is measured as generated tokens per second, aggregated over the benchmark execution period.
- **Memory Usage:** GPU memory consumption is sampled via NVIDIA NVML APIs, reporting peak and average usage during inference. This measurement captures the combined effects of model weights, activations, and serving-engine state (e.g., KV cache), reflecting practical deployment memory behavior.
- **Energy Consumption:** GPU power draw is sampled at fixed intervals during inference using NVIDIA GPU telemetry. Total energy consumption is computed by integrating power over time and subtracting an idle baseline

measured under matched system conditions. Energy is additionally normalized per query to enable fair comparison across benchmarks of differing sizes.

All measurements are collected under isolated GPU execution to minimize interference from concurrent workloads.

F. Evaluation Metrics

We report metrics in three categories:

- **Task Performance:** Exact-match accuracy is reported for MMLU and GSM8K, while normalized accuracy is used for HellaSwag. An unweighted mean across tasks is reported as a summary indicator.
- **Runtime Performance** Latency (mean, median, P95, P99) and throughput (tokens/sec).
- **Resource and Energy Efficiency:** Peak and average GPU memory usage, total energy consumption (kWh), average power draw (W), and energy per query (J/query).

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

IV. Experimental Results

A. Experimental Setup

We evaluate five instruction-tuned SLMs under three precision modes (FP16, INT8, INT4), yielding 13 evaluated model-precision configurations; Gemma-3-4B-it and one additional configuration were evaluated only in FP16 due to quantization compatibility constraints. All reported results are obtained on NVIDIA A100 (80GB) GPUs via Vast.ai. We performed initial pipeline validation and Weights & Biases integration testing on NVIDIA A100 (40GB) GPUs; these pilot runs are not used for reported comparisons.

Inference is executed using `vLLM` [26] with greedy decoding (temperature 0, top-p 1.0). Benchmarks (MMLU, GSM8K, HellaSwag) are evaluated using `lm-evaluation-harness` [25] with standardized prompting and scoring. GPU memory is tracked via NVML APIs and energy is computed by integrating GPU power telemetry with idle-baseline subtraction (Section III-F). In total, the evaluation consumed >200 GPU-hours and processed ∼800K inference samples.

B. Performance Comparison Before and After Quantization

1) Baseline Performance (FP16): Table II 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 (e.g., KV cache and activations) rather than weight storage alone.

2) Quantization Impact (INT8/INT4 vs FP16): Table III summarizes relative changes from FP16 baselines under INT8 and INT4 quantization.

INT8 provides consistent acceleration for most models ($1.32 - 2.18\times$) 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 ($0.96-1.63\times$) 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 ($2.18\times$), whereas Qwen2.5-3B exhibits no speedup under INT8/INT4 and the largest GSM8K degradation under INT4.

C. Analysis of Results

1) Pareto Frontier Analysis: Figure 1 summarizes multi-objective trade-offs across latency, memory, energy, and accuracy. No single configuration dominates across objectives, motivating deployment-dependent selection.

`pareto_frontiers.png`

Fig. 1. Pareto frontiers illustrating trade-offs across accuracy and efficiency dimensions for FP16/INT8/INT4 configurations.

Pareto analysis highlights distinct operating points: Phi-3-mini (FP16) maximizes accuracy (0.701) at moderate energy, Llama-3.2-3B (INT8) provides a balanced configuration (0.099 s latency, 0.605 avg accuracy, 35.3% energy reduction), and Llama-3.2-3B (INT4) minimizes energy (0.011 kWh) with reduced accuracy (0.569). These results emphasize that deployment decisions must explicitly trade accuracy against latency and energy rather than relying on single-metric rankings.

TABLE II
BASELINE PERFORMANCE IN FP16 (A100-80GB). LATENCY IS PER-QUERY.

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 III
QUANTIZATION IMPACT RELATIVE TO FP16 (POSITIVE SPEEDUP = FASTER; NEGATIVE MEMORY CHANGE = HIGHER MEMORY).

Model	Prec.	Speedup	Memory Change (%)	Energy Red. (%)	MMLU Drop (%)	GSM8K Drop (%)	HellaSwag Drop (%)
Mistral-7B	INT8	1.45×	0.00	45.0	-0.16	-5.77	-0.36
Mistral-7B	INT4	1.63×	-0.10	62.6	-1.98	-8.95	-0.73
Llama-3.2-3B	INT8	2.18×	-0.09	35.3	-0.05	-0.90	0.08
Llama-3.2-3B	INT4	1.63×	-0.03	49.1	-2.89	-11.30	-1.57
Phi-3-mini	INT8	1.32×	-0.05	45.7	-1.38	-9.03	-0.66
Phi-3-mini	INT4	1.01×	-0.05	50.6	-3.16	-9.98	-2.49
Qwen2.5-3B	INT8	0.99×	0.87	12.6	-1.20	-1.16	-1.55
Qwen2.5-3B	INT4	0.96×	0.02	35.0	-3.30	-18.48	-1.99

2) Recommended Configurations: Table IV summarizes recommended model–precision choices for common deployment constraints.

TABLE IV
RECOMMENDED CONFIGURATIONS BY DEPLOYMENT OBJECTIVE.

Objective	Recommended configuration
Accuracy-critical	Phi-3-mini (FP16)
Latency-critical	Llama-3.2-3B (INT8)
Energy-constrained	Llama-3.2-3B (INT4)

V. DISCUSSION

A. Interpretation of Results

Our evaluation yields several insights that challenge common assumptions about quantization and deployment of small language models (SLMs).

1) Model-Dependent Quantization Behavior: Quantization effectiveness varies substantially across models, with INT8 speedups ranging from $0.99\times$ to $2.18\times$. 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, attention implementations, and activation distributions, play a decisive role in quantization compatibility. The strong robustness observed in newer architectures suggests that some models may implicitly incorporate quantization-friendly design choices, while others may require architecture-specific or layer-aware quantization strategies.

2) Task Sensitivity and Cognitive Complexity: Quantization impacts tasks unevenly across all evaluated models. Mathematical reasoning (GSM8K) degrades $3-10\times$ more than factual knowledge (MMLU) and commonsense reasoning (HellaSwag) under both INT8 and INT4 schemes. This consistent pattern supports the hypothesis that multi-step arithmetic reasoning is particularly sensitive to numerical precision, as small quantization errors can accumulate across chained reasoning steps. In contrast, single-step retrieval and plausibility judgments remain comparatively robust. This finding has direct deployment implications: applications requiring arithmetic or logical reasoning should prioritize FP16 or conservative INT8 precision, while factual or classification tasks can tolerate more aggressive quantization.

3) Memory Behavior in Modern Serving Engines: Despite theoretical expectations of $2-4\times$ memory reduction from low-bit quantization, we observe negligible changes in end-to-end GPU memory usage across precision modes. This behavior arises because inference-time memory is dominated by KV-cache storage, activation tensors, and framework overhead rather than model weights alone. In our experiments, model weights account for only a small fraction of total GPU memory consumption. Consequently, memory-constrained deployments may benefit more from optimizing context length, batch size, and KV-cache management than from weight-only quantization.

B. Comparison with Previous Studies

1) Relation to SLM-Bench: SLM-Bench [2] provides broad coverage across many SLMs and datasets, emphasizing baseline accuracy and environmental impact. Our work complements this breadth with depth by systematically analyzing how quantization reshapes accuracy-efficiency trade-offs within representative SLM families. Consistent with SLM-Bench, we find that model size alone is a poor predictor of efficiency and that smaller models can outperform larger ones in practical settings. Beyond this, we demonstrate that quantization can substantially alter a model's deployment profile, enabling speedups and energy reductions without requiring model replacement.

2) Relation to HELM: HELM [3] introduced holistic evaluation across multiple dimensions, including limited efficiency metrics, primarily for large language models. Our work adopts a similar multi-dimensional philosophy but focuses explicitly on SLM deployment scenarios where efficiency is often a first-order concern. Unlike HELM, we treat quantization as a central axis of evaluation and construct Pareto frontiers over accuracy, latency, and energy. This narrower scope enables deeper analysis of deployment-critical trade-offs that are infeasible to explore at HELM's scale.

C. Practical Implications

Our findings indicate that SLM deployment decisions should explicitly account for both task requirements and quantization behavior. INT8 quantization emerges as a strong default for many production scenarios, particularly when applied to architectures such as Llama-3.2-3B that exhibit high quantization robustness. INT4 quantization can provide additional efficiency gains but should be reserved for tasks tolerant to accuracy loss or environments with strict energy constraints. Importantly, practitioners should not assume uniform quantization benefits across models; empirical evaluation remains essential.

D. Limitations

This study has several limitations. First, our evaluation covers five instruction-tuned SLMs, which, while representative, do not capture the full diversity of emerging architectures. Second, the benchmark suite emphasizes knowledge and reasoning tasks and does not include code generation, multilingual evaluation, or long-context understanding. Third, all experiments are conducted on NVIDIA A100 GPUs with ample memory, meaning memory-constrained scenarios may exhibit different trade-offs. Finally, energy measurements rely on GPU-level telemetry and do not capture system-wide or economic costs. These limitations motivate further investigation beyond the scope of this work.

VI. Conclusion

A. Summary of Findings

This work introduced *SLiM-Eval*, a systematic framework for evaluating small language models (SLMs) across accuracy and efficiency dimensions under post-training quantization. Through over 200 hours of controlled experimentation spanning 13 model–precision configurations, we empirically characterized how quantization reshapes real-world deployment trade-offs.

Our results show that INT8 quantization is a strong default optimization, delivering 1.3–2.2$\times$ inference speedup and 35–63% energy reduction while preserving 95–99% of FP16 accuracy across most models. Llama-3.2-3B emerged as particularly robust, achieving a 2.18$\times$ INT8 speedup with less than 1% accuracy loss, making it a compelling choice for balanced production deployment.

We further demonstrate that quantization sensitivity is task-dependent: mathematical reasoning (GSM8K) degrades 3–10$\times$ more than factual knowledge (MMLU) or commonsense reasoning (HellaSwag), indicating that conservative precision is preferable for reasoning-intensive workloads, while INT4 quantization remains viable for knowledge retrieval and classification. Across models, quantization effectiveness varies substantially, confirming that benefits cannot be inferred from parameter count or model family alone.

Finally, we find that end-to-end GPU memory savings from quantization are limited in modern serving engines. Despite theoretical 2–4$\times$ weight compression, observed memory usage varies by less than 1%, as inference memory is dominated by KV-cache and activation tensors, highlighting the need for holistic optimization beyond weight quantization alone.

B. Contributions

This work makes three primary contributions to the evaluation and deployment of small language models. (1) We provide the first systematic empirical characterization of post-training quantization trade-offs across accuracy, latency, throughput, memory, and energy, revealing strong task- and architecture-dependent effects. (2) We establish a reproducible, multi-dimensional evaluation methodology integrating standardized benchmarks, modern inference serving, quantization, and fine-grained efficiency profiling. (3) We demonstrate deployment-oriented, multi-objective analysis using Pareto frontiers and derive actionable guidance for accuracy-critical, latency-critical, energy-constrained, and balanced deployment scenarios.

C. Recommendations for Future Research

Our findings motivate several directions for future work. Expanding evaluation to additional SLM architectures and task domains (e.g., code generation, long-context reasoning, multilingual benchmarks) would improve generalizability. Investigating alternative quantization methods, including AWQ, SmoothQuant, mixed-precision schemes, and quantization-aware fine-tuning, may improve robustness for quantization-sensitive tasks.

The strong architecture dependence observed suggests opportunities for architecture–quantization co-design and quantization-aware pre-training. Future studies should also bridge controlled benchmarks with real-world serving by examining multi-turn dialogue, KV-cache reuse, dynamic batching, and request routing across quantization variants, as well as extending energy measurements toward holistic lifecycle and environmental impact analysis.

References

[1] Y. Lu *et al.*, "Small Language Models: Survey, Measurements, and Insights," *arXiv preprint arXiv:2409.15790*, 2024.

[2] N. T. Pham, T. Kieu, D.-M. Nguyen, S. H. Xuan, N. Duong-Trung, and D. Le-Phuoc, "SLM-Bench: A comprehensive benchmark of small language models on environmental impacts—Extended version," *arXiv preprint arXiv:2508.15478v2*, 2025.

[3] P. Liang *et al.*, "Holistic evaluation of language models," *arXiv preprint arXiv:2211.09110*, 2023.

[4] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[5] X. Jiao *et al.*, "TinyBERT: Distilling BERT for natural language understanding," in *Findings of the Association for Computational Linguistics: EMNLP*, pp. 4163–4174, 2020.

[6] Y. Li *et al.*, "Textbooks are all you need II: phi-1.5 technical report," *arXiv preprint arXiv:2309.05463*, 2023.

[7] Meta AI, "Llama 3.2: Revolutionizing edge AI and vision with open, customizable models," Meta AI Blog, 2024. [Online]. Available: https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/

[8] A. Wang *et al.*, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *Proc. EMNLP Workshop BlackboxNLP*, pp. 353–355, 2018.

[9] A. Wang *et al.*, "SuperGLUE: A stickier benchmark for general-purpose language understanding systems," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, pp. 3266–3280, 2019.

[10] X. Wu, C. Li, R. Y. Aminabadi, Z. Yao, and Y. He, "Understanding INT4 Quantization for Transformer Models: Latency Speedup, Composability, and Failure Cases," *arXiv preprint arXiv:2301.12017*, 2023.

[11] B. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE CVPR*, pp. 2704–2713, 2018.

[12] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," in *Proc. ICLR*, 2023.

[13] E. Frantar and D. Alistarh, "Optimal brain compression: A framework for accurate post-training quantization and pruning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 4475–4488, 2022.

[14] J. Lin *et al.*, "AWQ: Activation-aware weight quantization for LLM compression and acceleration," *arXiv preprint arXiv:2306.00978*, 2023.

[15] G. Xiao *et al.*, "SmoothQuant: Accurate and efficient post-training quantization for large language models," in *Proc. ICML*, pp. 38087–38099, 2023.

[16] T. Dettmers *et al.*, "QLoRA: Efficient finetuning of quantized LLMs," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, pp. 10088–10115, 2023.

[17] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," in *Proc. ACL*, pp. 3645–3650, 2019.

[18] D. Patterson *et al.*, "Carbon emissions and large neural network training," *arXiv preprint arXiv:2104.10350*, 2021.

[19] V. Schmidt *et al.*, "CodeCarbon: Estimate and track carbon emissions from machine learning computing," 2021. [Online]. Available: https://codecarbon.io

[20] J. You, J.-W. Chung, and M. Chowdhury, "Zeus: Understanding and optimizing GPU energy consumption of DNN training," in *Proc. USENIX NSDI*, pp. 119–139, 2023.

[21] A. Recasens, S. Chintala, and R. Fergus, "Towards Pareto optimal throughput in small language model serving," in *Proc. EuroMLSys*, pp. 95–102, 2024.

[22] D. Hendrycks *et al.*, "Measuring massive multitask language understanding," in *Proc. ICLR*, 2021.

[23] K. Cobbe *et al.*, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021.

[24] R. Zellers *et al.*, "HellaSwag: Can a machine really finish your sentence?" in *Proc. ACL*, pp. 4791–4800, 2019.

[25] L. Gao *et al.*, "A framework for few-shot language model evaluation," Zenodo, 2023. [Online]. Available: https://doi.org/10.5281/zenodo.10256836

[26] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with PagedAttention," in *Proc. ACM SIGOPS Symp. Operating Systems Principles (SOSP)*, 2023.

Appendix

A. Software Environment and Core Dependencies

All experiments were conducted in a controlled Python-based environment with pinned library versions to ensure reproducibility. Rather than enumerating the full dependency tree (which includes transitive and system-level packages), we report here the core libraries directly relevant to model inference, quantization, evaluation, and profiling.

- **Deep Learning Framework:** PyTorch 2.8.0 with CUDA 12.8
- **Inference Engine:** `vLLM` 0.11.0 [26]
- **Evaluation Framework:** `lm-evaluation-harness` 0.4.x [25]
- **Quantization Toolkit:** `llmcompressor` 0.7.1 (GPTQ-based INT8 and INT4)
- **Model Hub and Tokenization:** `transformers` 4.55.2, `huggingface-hub` 0.36.0
- **Monitoring and Profiling:** NVIDIA NVML via `pynvml`, `nvidia-smi`
- **Experiment Tracking:** Weights & Biases (`wandb`)
- **Data Processing and Analysis:** `numpy` 2.2.6, `pandas` 2.3.3
- **Visualization:** `matplotlib` 3.10.7, `seaborn` 0.13.2

The complete dependency specification, including transitive packages and system utilities, is available in the project repository to support full environment reconstruction.

B. Detailed Experimental Configuration

This section provides additional experimental details omitted from the main paper for brevity.

1) Inference Configuration: All models are evaluated using greedy decoding with temperature set to 0.0 and top-p set to 1.0. The maximum context length is fixed at 8192 tokens, and the maximum generation length is capped at 256 tokens. GPU memory utilization is set to 0.9 to balance throughput and stability.

Batch sizes are task-dependent to reflect typical evaluation practice:

- MMLU: batch size 1
- GSM8K: batch size 32
- HellaSwag: batch size 32

2) Reproducibility Controls: To reduce variance and ensure reproducibility, we apply the following controls across all experiments:

- Fixed random seed (42) for all runs
- Deterministic CUDA operations enabled
- Three independent runs per configuration, with mean values reported
- Cache warm-up using 10 preliminary queries prior to measurement
- Idle cooling period of 60 seconds between experiments for thermal stabilization

C. Extended Quantization Results

This appendix provides extended quantitative results supporting the summary analyses presented in Section IV. Specifically, it includes full per-task accuracy values for all evaluated model–precision configurations, enabling detailed inspection beyond the aggregate metrics shown in the main paper.

Due to space constraints, only summarized performance comparisons (baseline FP16 and relative quantization impact) are reported in the main text. The extended tables include:

- Per-task accuracies (MMLU, GSM8K, HellaSwag) for FP16, INT8, and INT4
- Corresponding latency, throughput, and energy measurements for each configuration

These results confirm the trends discussed in the main paper, including task-specific sensitivity to quantization and architecture-dependent efficiency gains.

D. Quantization Compatibility Notes

Certain model configurations could not be evaluated under INT8 or INT4 precision due to incompatibilities with the quantization pipeline or serving engine at the time of experimentation. In particular, Gemma-3-4B-it and one additional configuration variant are reported only under FP16 precision.

These exclusions are purely technical and do not reflect model quality. All reported FP16 results for these models follow the same evaluation protocols as other configurations and are included to provide fair baseline comparisons.

E. Energy Measurement Methodology

GPU energy consumption is measured by integrating instantaneous GPU power draw sampled at 100 ms intervals via NVIDIA NVML. To isolate model-specific energy usage, an idle baseline is measured prior to each experiment and subtracted from total consumption. Reported energy values therefore reflect net inference energy attributable to model execution only.

F. Pareto Frontier Construction

Pareto frontiers are constructed by identifying non-dominated model-precision configurations under paired objectives (e.g., latency vs. memory, energy vs. accuracy). A configuration is considered Pareto-optimal if no other configuration achieves strictly better performance on both axes simultaneously.

G. Artifact Availability

The SLiM-Eval framework, experiment configurations, and analysis scripts will be released as open-source artifacts to support independent verification and extension of this work.