QUANTIZATION FOR REASONING PRESERVATION IN SMALL LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) have demonstrated exceptional capabilities in textual inference, but their practical deployment is constrained by high computation and memory overhead. To address this, we propose LLM-QRP (Quantization for Reasoning Preservation), a layer-aware, mixed-precision quantization framework designed to reduce model memory footprint while preserving multi-step reasoning capabilities. Unlike standard uniform post-training quantization methods that treat all parameters equally, LLM-QRP identifies and protects critical "thinking" sub-components by analyzing layer-wise Self-Logits Evolution Decoding (SLED) and entropy dynamics across the residual stream. These signals are unified via Principal Component Analysis (PCA) into a data-driven reasoning importance score. We then formulate precision allocation as a 0-1 Multiple-Choice Knapsack Problem solved via Integer Linear Programming (ILP) to assign higher bit-widths to high-sensitivity components under strict VRAM targets. Experiments on sub-500M parameter models across GSM8K, TruthfulQA, and MMLU demonstrate that LLM-QRP achieves an optimal Pareto frontier between model compression and task accuracy, maintaining near-baseline reasoning performance while outperforming uniform 8-bit baselines and alternative quantization methods.

1 INTRODUCTION

Large Language Models (LLMs) have qualitatively transformed various domains of artificial intelligence. A core advantage of local LLMs is the benefits of on-device use. It serves as an alternative pathway for LLMs to operate effectively offline, without intermediate delays from API or cloud calls, and remains beneficial for real-time, offline applications involving autonomous vehicles, financial processing, and applications requiring a fine-tuned model. However, with great model size comes a great compute cost.

In post-training-quantization (PTQ), multi-step reasoning tasks are compromised first, leading to degradation in the model's ability to reason and think. Quantization-aware training suffers from high infrastructure costs, making compute and model training quite expensive. Recent advancements in quantization provided by GPTQ and LLM-AWQ do provide mixed-precision quantization; LLM-AWQ optimizes for which weights to preserve instead of identifying layers that actually perform reasoning (Li et al., 2026).

Existing mixed-precision techniques attempt to mitigate accuracy degradation, but they primarily allocate precision based on weight-level reconstruction errors, activation outliers, or sensitivity metrics that fail to capture functional reasoning behavior across the network. We hypothesize that transformer layers do not contribute equally to reasoning. Instead, specific "thinking layers" drive the core steps of step-by-step logic and semantic refinement, whereas other layers perform lower-level processing or information routing. Uniformly compressing these reasoning-critical stages degrades the network's capacity to preserve logical coherence.

To address these limitations, we present LLM-QRP (Quantization for Reasoning Preservation), a layer-aware, mixed-precision framework designed to protect reasoning capabilities in small LLMs. Rather than treating quantization as an isolated weight-reconstruction problem, LLM-QRP identifies reasoning-critical layers during active Chain-of-Thought generation and preserves their precision accordingly.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2 BACKGROUND

Recent work on LLM quantization has explored a range of approaches for reducing memory and computational requirements while minimizing the degradation introduced by reduced numerical precision. These methods include weight-only quantization, activation-aware quantization, mixed-precision quantization, and methods that explicitly identify sensitive weights or layers. Our approach builds upon these developments while focusing specifically on the preservation of reasoning through layer-wise importance estimation.

2.1 MIXED-PRECISION QUANTIZATION

Several works have explored mixed-precision strategies to allocate different bit-widths to different components of an LLM. SpQR introduces a sparse-quantized representation that combines low-bit quantization with a small set of higher-precision outlier weights, allowing substantial memory compression while preserving model quality (Dettmers et al., 2023b). This demonstrates that selectively allocating additional precision to sensitive components can provide a favorable accuracy–memory trade-off. However, SpQR primarily operates through weight-level outlier identification, whereas LLM-QRP determines precision allocation using layer-level reasoning importance.

AMQ explores adaptive mixed-precision quantization by allocating different precision levels according to the sensitivity of model components (Lee et al., 2025). Such approaches demonstrate that a uniform quantization configuration is not necessarily optimal across an entire model. LLM-QRP extends this principle by using SLED and layer-wise entropy sensitivity to explicitly estimate the reasoning importance of individual transformer layers before assigning their precision. Unlike AMQ, which primarily optimizes the allocation of precision according to quantization sensitivity and efficiency objectives, LLM-QRP explicitly incorporates reasoning behavior into the precision-allocation process, allowing the resulting mixed-precision configuration to prioritize layers that contribute most strongly to reasoning preservation.

Slim-LLM further investigates mixed-precision quantization by identifying a small subset of important weights and allocating precision non-uniformly to improve the accuracy–compression trade-off (Huang et al., 2025). Similar to these approaches, LLM-QRP seeks a non-uniform precision allocation; however, our method differs in that the allocation is guided by a reasoning-oriented layer importance score rather than solely by weight reconstruction or quantization sensitivity.

2.2 QUANTIZATION AND REASONING PRESERVATION

More recent studies have demonstrated that quantization can disproportionately affect reasoning-intensive capabilities, particularly under aggressive low-bit quantization (Li et al., 2026). This motivates moving beyond conventional reconstruction-error objectives and explicitly considering the capabilities that should be preserved during quantization.

Unlike prior methods that evaluate whole transformer blocks, LLM-QRP profiles model dynamics at fine-grained sub-component granularity (Q, K, V, O projections and MLP sub-blocks). LLM-QRP addresses this gap by introducing a layer-aware reasoning preservation framework. We combine Self-Logits Evolution Decoding (SLED), which characterizes the evolution of intermediate predictions toward the final output distribution (Zhang et al., 2025), with layer-wise entropy sensitivity to construct a unified reasoning importance score. These metrics are aggregated using unsupervised Principal Component Analysis (PCA) to derive a data-driven Reasoning Sensitivity Score ($R_{l,c}$).

Using these scores, LLM-QRP formulates bit assignment as a 0-1 Multiple-Choice Knapsack Problem (MCKP). Solved globally via Integer Linear Programming (ILP), it maximizes retained reasoning sensitivity subject to strict target bit budgets (B_{target}).

In contrast to existing approaches such as AWQ, SpinQuant, SpQR, and Slim-LLM, which primarily optimize weight representation, activation distributions, outliers, or quantization error, LLM-QRP explicitly considers the *functional role of individual transformer layers in reasoning*. Our framework is therefore complementary to existing quantization engines: existing quantization methods can be used to perform the underlying weight compression, while LLM-QRP determines where higher or lower precision should be allocated based on reasoning importance.

3 QUANTIZATION FOR REASONING PRESERVATION IN SMALL LLMS

We observe that not all layers contribute equally to the model's final response, and there exists an optimal distribution of quantized layers such that the model retains most of its efficacy whilst maintaining a significantly smaller memory footprint. The intuition behind this is to find the "*thinking layers*" (Kapl et al., 2025).

3.1 LOGITS EVOLUTION ACROSS TRANSFORMER LAYERS

Modern large language models are architecturally composed of N model layers, with a vocabulary space defined as $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$. The first metric compares the KL divergence between the distributions at adjacent layers L_i and L_{i+1} by projecting high-dimensional hidden states back to the vocabulary space and converting them into logit values using the LM head W. A Softmax is then applied to these logits, converting them into probability distributions. To measure the "thinking" occurring between adjacent layers, we calculate the KL divergence between their corresponding output distributions.

$$\boldsymbol{\ell}_i = \underbrace{W\mathbf{h}_i + \mathbf{b}}_{\text{LM head projection}}, \qquad P_i(v) = \underbrace{\frac{\exp(\ell_i(v))}{\sum_{u\in\mathcal{V}} \exp(\ell_i(u))}}_{\text{probability distribution over vocabulary}} \tag{1}$$

$$D_{KL}(P_i \| P_{i+1}) = \underbrace{\sum_{v\in\mathcal{V}} P_i(v) \log\left(\frac{P_i(v)}{P_{i+1}(v)}\right)}_{\text{distributional change between adjacent layers}} \tag{2}$$

After applying a standard KL divergence analysis, we can use SLED-specific methods, which compute a weighted average of these intermediate layers, with layers that contain more "accurate" signals given greater influence on the final output, more accurately representing knowledge transfer between latent layers. Here, the core intuition is that a *"thinking layer*" is not just one that changes, but one whose change aligns semantically with the final output. First, we define a set of candidate premature layers at a fixed stride Δ, as choosing each layer is too computationally expensive. We can then project both candidate layers and final mature layers into logit values (Zhang et al., 2025).

$$\mathcal{L}_{\text{cand}} = \underbrace{\{L_1, L_{1+\Delta}, L_{1+2\Delta}, \ldots, L_N\}}_{\text{candidate layers evaluated at fixed stride}}, \qquad \boldsymbol{\ell}_i = \underbrace{W\,\text{Norm}(\mathbf{h}_i)}_{\text{candidate layer logits}}, \quad L_i \in \mathcal{L}_{\text{cand}} \tag{3}$$

$$\boldsymbol{\ell}_M = \underbrace{W\,\text{Norm}(\mathbf{h}_N)}_{\text{mature output logits}}, \qquad \mathbf{g}_{i,t} = \underbrace{\text{softmax}(\boldsymbol{\ell}_i) - \mathbf{1}_t}_{\text{evolution gradient}}, \qquad \boldsymbol{\delta}_i = \underbrace{\log\text{softmax}(\boldsymbol{\ell}_i) - \log\text{softmax}(\boldsymbol{\ell}_M)}_{\text{logit divergence from mature output}} \tag{4}$$

Let $\boldsymbol{\ell}_M$ denote the mature output. A candidate mask is constructed using a relative log-probability threshold (e.g., 0.1) from the mature output, eliminating any vocabulary noise. For each candidate layer, we define the Evolution Gradient relative to the top-k tokens using a one-hot token method and the Logit Divergence.

To compute fine-grained divergence, intermediate hidden representations are first projected into the vocabulary space using logit-lens operations on the residual stream outputs of the Attention and MLP sub-components:

$$l_{l,t}^{\text{attn}} = W_{\text{lm_head}} \cdot \text{Norm}(h_{l,t}^{\text{attn}}), \quad l_{l,t}^{\text{mlp}} = W_{\text{lm_head}} \cdot \text{Norm}(h_{l,t}^{\text{mlp}})$$

For each sub-component $c \in \{\text{attn}, \text{mlp}\}$, a candidate mask is constructed using a relative log-probability threshold from the mature output ℓ_M to filter out vocabulary noise. We then define the Evolution Gradient $g_{l,t}^{(c)}$ relative to the top-k tokens and compute the intermediate Logit Divergence $\delta_{l,t}^{(c)} = \log\text{softmax}(l_{l,t}^{(c)}) - \log\text{softmax}(\ell_M)$

The final Shannon Logit Evolution Dynamics (S_{SLED}) score for sub-component c in layer l is calculated as the average cosine similarity between evolution gradients and logit divergence across all reasoning tokens:

$$S_{\text{SLED}}(l, c) = \frac{1}{|\mathcal{T}_{\text{CoT}}|} \sum_{t \in \mathcal{T}_{\text{CoT}}} \cos\left(g_{l,t}^{(c)}, \delta_{l,t}^{(c)}\right)$$

3.2 Layer-wise Entropy Profiling

The concept of entropy is applied to gauge an LLM's uncertainty as it moves through the residual stream, thereby defining its probabilistic convergence. While a model converts tokens to natural language at the language modelling head, a probability distribution is updated at every layer; each intermediate distribution can be calculated by passing the layer's state through a logit lens (Wang, 2025). This involves forcing the middle layer's hidden state through the logit lens (Huang et al., 2025), depicting the model's uncertainty at layer i as a projection of latent representations in probability space. Ideally, the layers transition from high-entropy (semantic exploration) to low-entropy (probabilistic convergence) as the model converges upon a thought; each layer contributes a marginal KL-divergence towards the final logit distribution (Zhao, 2026).

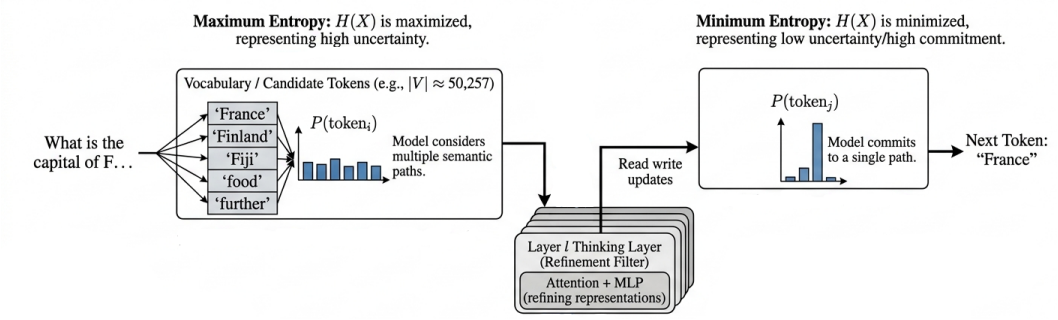

Figure 1: Probabilistic convergence through the residual stream. High-entropy exploration in early layers transitions to low-entropy convergence in final reasoning layers.

While S_{SLED} quantifies output distributional alignment, it does not measure how aggressively a layer reduces token ambiguity during forward propagation. The layers with the highest quantization sensitivity are the most vital to the inference process; the model's performance will fall when they are lowered in precision as compared to less sensitive layers.

The ΔH metric is implemented algorithmically, beginning with a forward pass in BFloat16 (BF16) precision to determine the output's original entropy and distribution. Subsequently, a layer-wise quantization sensitivity test is conducted: for each transformer block i within the multi-head attention and feed-forward network projection, a 4-bit quantization is injected. The Shannon entropy of this modified distribution is then computed as the uncertainty of a model's output distribution P over the vocabulary $\mathcal{V}$.

$$H(P_{l,t}^{(c)}) = \underbrace{-\sum_{v \in \mathcal{V}} P_{l,t}^{(c)}(v) \log P_{l,t}^{(c)}(v)}_{\text{Vocabulary Uncertainty}} \quad \text{where} \quad \underbrace{P_{l,t}^{(c)} = \text{softmax}(l_{l,t}^{(c)})}_{\text{Projected Distribution}}$$

Sub-component Entropy Velocity measures the absolute information shift between the input residual stream $H(P_{l,t}^{\text{in}})$ and output residual stream $H(P_{l,t}^{\text{out}})$ across the sub-block:

$$\Delta H(l, c) = \frac{1}{|\mathcal{T}_{\text{CoT}}|} \sum_{t \in \mathcal{T}_{\text{CoT}}} \left| H(P_{l,t}^{\text{in}}) - H(P_{l,t}^{\text{out}}) \right|$$

Sub-components exhibiting high ΔH indicate active state transitions where prediction confidence is rapidly sharpened, marking them as sensitive computation bottlenecks during Chain-of-Thought generation.

3.3 MULTI-SIGNAL PCA AGGREGATION

Rather than combining $S_{\text{SLED}}(l,c)$ and $\Delta H(l,c)$ using arbitrary scalar weights (λ), we employ Principal Component Analysis (PCA) to derive optimal, data-driven feature weights directly from model activation dynamics. First, $S_{\text{SLED}}(l,c)$ and $\Delta H(l,c)$ metrics are standardized across all sub-components into zero-mean, unit-variance vectors $\tilde{S}_{\text{SLED}}$ and $\tilde{\Delta H}$. We form the sub-component observation matrix $\mathbf{X} \in \mathbb{R}^{2L\times 2}$:

$$\mathbf{x}_{l,c} = \left[\tilde{S}_{\text{SLED}}(l,c),\ \tilde{\Delta H}(l,c)\right]$$

We extract the dominant eigenvector $\mathbf{w} = [w_1, w_2]^T$ corresponding to the largest eigenvalue of the empirical covariance matrix $\mathbf{X}^T\mathbf{X}$. The unified Reasoning Sensitivity Score $R_{l,c}$ is defined as the projection onto the first principal component (PC_1):

$$R_{l,c} = w_1 \cdot \tilde{S}_{\text{SLED}}(l,c) + w_2 \cdot \tilde{\Delta H}(l,c)$$

This unsupervised formulation eliminates manual hyperparameter tuning while maximizing signal variance, guaranteeing that sub-components driving core reasoning capabilities receive systematically higher sensitivity rankings.

3.4 MIXED-PRECISION QUANTIZATION FRAMEWORK

We employ a Mixed-Precision Quantization (MPQ) framework to convert the reasoning importance scores achieved through layer ablation and SLED/entropy analyses into a tangible structure for model deployment (Nagel et al., 2021; Gholami et al., 2021). This strategy partitions model layers and allocates precision between them rather than quantizing models all in one precision (Huang et al., 2025). While quantization is bound to reduce performance due to the nature of compressing model weights (Frantar et al., 2023), mixed-precision allows for a greater model accuracy-to-size ratio by compressing only certain layers (Xiao et al., 2024), (Dettmers et al., 2022). Transformer models do not act as a monolithic structure but rather as a pipeline of stages (Vaswani et al., 2023), passing information and transforming it as it traverses the residual stream (Kapl et al., 2025). Some of these stages are more thinking-intensive, while others just relay the information with little change; a mixed-precision strategy works well to respect this downstream architecture (Harma et al., 2026).

3.5 REASONING-GUIDED LAYER ALLOCATION

After the reasoning importance scores are computed, they are used to rank each layer's contribution to the final output. The lowest-ranked layers are then quantized appropriately, and their precision is instead allocated towards the higher importance layers (Yao et al., 2022). Once layers are ranked, the bottom percentile is assigned to FP4, the middle percentile to INT8, and the remainder preserved in BF16, or full precision (Dettmers et al., 2022). This algorithm runs to find a Pareto Front for quantization (Huang et al., 2025), where the bit-rate has a significant drop while retaining accuracy above a defined floor, $\alpha_{\min}$ (Cobbe et al., 2021; Zhang et al., 2024).

To determine optimal bit allocations under strict hardware constraints, we formulate precision assignment as a 0-1 Multiple-Choice Knapsack Problem (MCKP), removing heuristic percentile grid sweeps.

Let $x_{l,c,b} \in \{0,1\}$ be a binary decision variable indicating whether sub-component (l,c) is assigned bit-width $b \in \{2,3,4,8,16\}$. Given a target average bit budget B_{target} (e.g., 3.5 bits per parameter across the entire model), the optimal allocation is obtained by solving:

$$\max_{\{x_{l,c,b}\}} \sum_{l=1}^{L} \sum_{c} \sum_{b} x_{l,c,b} \cdot R_{l,c} \cdot f(b)$$

$$\text{subject to} \quad \frac{1}{P_{\text{total}}} \sum_{l=1}^{L} \sum_{c} \sum_{b} x_{l,c,b} \cdot P_{l,c} \cdot b \le B_{\text{target}}$$

$$\sum_{b} x_{l,c,b} = 1 \quad \forall (l,c), \quad x_{l,c,b} \in \{0,1\}$$

where $P_{l,c}$ is the parameter count of sub-block (l, c), P_{total} is total model parameters, and $f(b)$ represents the non-linear precision fidelity function measuring relative information preservation at bit-width b.The Integer Linear Program (ILP) is solved globally using standard branch-and-bound algorithms in sub-second runtime, producing an exact, mathematically optimal precision assignment that maximizes retained reasoning capacity for any specified memory footprint B_{target}.

To safeguard structural integrity under aggressive compression, components with sensitivity $R_{l,c} \geq 0.75$ (along with a mandatory top-5% defensive floor of highest-criticality components) are hard-capped to ≥ 8-bit precision, while global macro cap restricts allocation to $\leq 8\%$ at 3-bit and $\leq 25\%$ at ≤ 6-bit. Furthermore, the ILP objective replaces raw bit-widths with an outlier-adjusted effective bit-width $b_{\text{eff}} = b + \epsilon(16 - b)$ (where $\epsilon = 0.001$) to explicitly account for non-weight metadata overhead during optimization.

4 Experiments and Benchmarking

All models are decoder-only causal language models loaded from the HuggingFace Hub in BF16 precision. We evaluate four sub-500M parameter models: (1) SmolLM2-135M (Allal et al., 2025), (2) Granite-4.0-350M (IBM Research, 2026), (3) Qwen2.5-0.5B (Bai et al., 2023), and (4) LFM2-350M (Amini et al., 2025). SmolLM2-135M is used as the primary model for detailed analysis, while the remaining models are used to evaluate the consistency of LLM-QRP across different architectures.

We evaluate the models on three benchmarks: GSM8K (Cobbe et al., 2021), TruthfulQA (Lin et al., 2022), and MMLU (Hendrycks et al., 2021). GSM8K serves as the primary reasoning benchmark, while TruthfulQA and MMLU provide complementary evaluations of factuality and general knowledge. Unless otherwise stated, 50 samples are evaluated per dataset. GSM8K uses chain-of-thought prompting, while TruthfulQA and MMLU use direct-answer prompts. To ensure statistical reliability, evaluations are conducted over full test splits across 5 random seeds, reporting mean accuracy and standard deviation (μ_σ). GSM8K uses 8-shot chain-of-thought prompting, while TruthfulQA and MMLU utilize standard 5-shot direct-answer prompts.

Experiments are implemented using PyTorch 2.5.1 with CUDA 12.1. Quantization experiments use bitsandbytes for uniform INT8 and INT4/FP4 quantization. GPTQ, AWQ, SpQR, SliM-LLM, SmoothQuant, and Atom are evaluated using their respective implementations. Calibration-based quantization methods use 32 sequences of up to 2048 tokens sampled from the GSM8K training split. SLED and entropy profiling are performed on CUDA-enabled hardware.

The main benchmark compares BF16, uniform INT8, uniform INT4, GPTQ, AWQ, SpQR, SliM-LLM, SmoothQuant, Atom, and LLM-QRP.

To ensure a strictly fair comparison, all dynamic mixed-precision baselines and LLM-QRP configurations are evaluated under identical target memory constraints ($B_{\text{target}} \in \{3.0, 3.5, 4.0\}$ bits per parameter).

We evaluate two small open checkpoints loaded in BF16, SmolLM2-135M (Allal et al., 2025) and Qwen2.5-0.5B (Bai et al., 2023), profiled on GSM8K chain-of-thought traces, against uniform INT8, uniform INT4, and released implementations of GPTQ (Frantar et al., 2023) , AWQ (Lin et al., 2026), SpQR (Dettmers et al., 2023b), SliM-LLM (Huang et al., 2025), SmoothQuant (Xiao et al., 2023), and Atom (Zhao et al., 2024) calibrated on GSM8K. Exact-match accuracy on GSM8K is near zero for models this small, so we report exp(NLL), the mean probability assigned to the ground-truth answer tokens, averaged over 5 seeds with 50 held-out examples per seed; we call it retention when normalized against the BF16 baseline. The profile is computed once from chain-of-thought traces on held-out GSM8K training sequences and then frozen, so nothing is tuned on the evaluation split. QRP allocates bits per sub-component from the criticality score $R_{l,c}$ (SLED disagreement and convergence velocity H fused by PCA), keeping the most critical components and the top-0.1

4.1 Main Results

Table 1 reports the comparison. QRP is near-lossless at roughly half the footprint on both models: SmolLM2-135M retains 99.6% of its BF16 performance (0.1996 vs. 0.2005, at 103.0 MB vs. 212.3 MB), and Qwen2.5-0.5B retains the full 100.0% (0.4939 vs. 0.4938, at 352.6 MB vs. 715.7 MB).

On TruthfulQA and MMLU the absolute numbers are small and QRP matches its BF16 base within one standard deviation, where uniform low-bit methods degrade. On GSM8K QRP is the best non-BF16 method on both models: 0.1996 against the next-best 0.1991 (SmoothQuant) on SmolLM2-135M, and 0.4939 against 0.4933 (SmoothQuant) on Qwen2.5-0.5B. Uniform INT8, the nearest single-precision configuration in size, is both larger than QRP (106.2 vs. 103.0 MB; 357.8 vs. 352.6 MB) and weaker on GSM8K (0.1960 vs. 0.1996; 0.4845 vs. 0.4939). QRP's average bit-width sits just under this operating point ($B_{\text{avg}} = 7.76$ on SmolLM2-135M, 7.88 on Qwen2.5-0.5B), yet it holds BF16-level retention there, where uniform INT8 retains only $\sim 98\%$ of the BF16 probability mass. On SmolLM2-135M no configuration exceeds QRP's 0.1996 except the BF16 base itself (0.2005, at $2.06\times$ the footprint); on Qwen2.5-0.5B even the BF16 base does not (0.4939 vs. 0.4938).

Table 1: Primary comparison across both models and all quantization methods (BF16, uniform INT8/INT4, GPTQ, AWQ, SpQR, SliM-LLM, SmoothQuant, Atom, and LLM-QRP): GSM8K, TruthfulQA, and MMLU retained probability $\exp(-\text{NLL})$ (5-seed means), average effective bits-per-weight, and on-disk footprint.

MODEL	METHOD	B_{avg} (bits)	FOOTPRINT (MB)	GSM8K	TRUTHFULQA	MMLU
SMOLLM2-135M	BF16	16.0	212.3	0.2005	0.0455	0.0080
	Uniform INT8	8.0	106.2	0.1960	0.0448	0.0082
	Uniform INT4	4.0	53.1	0.1528	0.0359	0.0051
	GPTQ (4-bit)	4.0	53.1	0.1471	0.0260	0.0038
	AWQ (4-bit)	4.0	53.1	0.1784	0.0412	0.0065
	SpQR (3-bit)	3.0	55.7	0.1834	0.0430	**0.0082**
	SliM-LLM (2-bit)	2.0	26.5	0.0007	0.0000	0.0000
	SmoothQuant (8-bit)	8.0	106.2	0.1991	**0.0452**	0.0080
	Atom (4-bit)	4.0	53.1	0.1199	0.0298	0.0042
	LLM-QRP	7.76	103.0	**0.1996**	0.0439	0.0080
QWEN2.5-0.5B	BF16	16.0	715.7	0.4938	0.0340	0.0065
	Uniform INT8	8.0	357.8	0.4845	0.0343	0.0059
	Uniform INT4	4.0	178.9	0.4022	**0.0354**	**0.0085**
	GPTQ (4-bit)	4.0	178.9	0.3948	0.0239	0.0045
	AWQ (4-bit)	4.0	178.9	0.4531	0.0350	0.0072
	SpQR (3-bit)	3.0	188.5	0.4855	0.0338	0.0062
	SliM-LLM (2-bit)	2.0	89.5	0.0307	0.0001	0.0001
	SmoothQuant (8-bit)	8.0	357.8	0.4933	0.0341	0.0065
	Atom (4-bit)	4.0	178.9	0.3873	0.0278	0.0059
	LLM-QRP	7.88	352.6	**0.4939**	0.0333	0.0064

4.2 EFFICIENCY AND DEPLOYMENT

At this allocation, per-MB efficiency

$$\eta = \frac{\exp(-\text{NLL})_{\text{GSM8K}}}{\text{footprint (MB)}}$$

roughly doubles relative to BF16 on both models (SmolLM2 $2.05\times$, Qwen $2.03\times$) while retaining $\geq 99.6\%$ of BF16 probability mass. Relative to uniform INT4, QRP gives up some per-MB efficiency (INT4 runs at ~ 1.5–$1.6\times$ QRP's η), but it buys near-lossless retention: INT4 loses a fifth to a quarter of the BF16 probability mass (76.2% and 81.4% retained). In size terms QRP sits within a few percent of uniform INT8 and roughly twice uniform INT4's size, and it is the smallest configuration on either model that reaches BF16-level retention: the uniform knobs cannot get there, since INT8 tops out at $\sim 98\%$ and INT4 trades a fifth to a quarter of the probability mass for a $4\times$ size cut. QRP optimizes memory, not latency; its on-device peak VRAM and per-token latency sit between uniform INT8 and INT4.

4.3 RETENTION ABLATION

To test whether the profiling signal predicts which sub-components are safe to quantize, we quantize a 20% sub-component subset to 4-bit (the rest stay BF16) and measure GSM8K retention over 5 seeds, selecting the subset by each candidate signal as bottom (least active), top (most active), or random (control), plus an all-components uniform reference. SmolLM2-135M separates the signals

cleanly; Qwen2.5-0.5B stays above 97% under every rule, including the uniform 4-bit reference, so its sub-components tolerate aggressive quantization uniformly and we base the ranking claims below on SmolLM2-135M.

On SmolLM2-135M uniform 4-bit drops retention to 76.9%, while every selective 20% scheme keeps 93% or more. The convergence velocity ΔH is the operative signal: quantizing the least active fifth retains 96.3% of BF16, whereas quantizing the most active fifth falls to 93.2% (random selection averages 96.3%). SLED (95.1% vs. 93.8%) and the fused criticality $R = \mathrm{PCA}(\mathrm{SLED}, \Delta H)$ (95.1% vs. 94.0%) separate these groups in the same direction, so the profile trusts ΔH for the low-bit selection step.

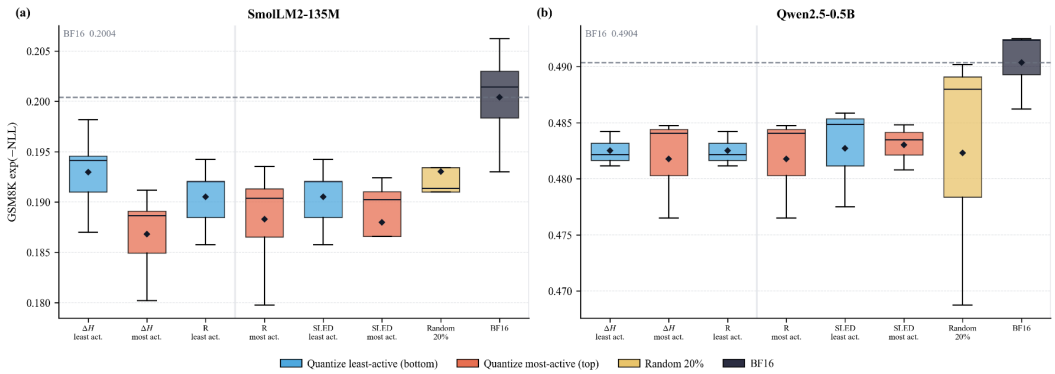

Figure 2: Distribution of GSM8K accuracy retention across profiling selection strategies on SmolLM2-135M and Qwen2.5-0.5B.

5 Limitations

The primary limitation of the provided quantization method is the high computational expense for larger models. Most of the experiments in this paper were performed on consumer-grade graphics cards, specifically the NVIDIA GeForce RTX 4070, which may neglect performance or scalability in training larger models or on different hardware. Additionally, LLM-QRP's evaluation relies on a limited set of datasets, which may not adequately capture edge cases or provide sufficient diversity in sample distribution. For instance, there is strong evidence that GSM8K test examples have leaked into the training data of many Large Language Models (LLMs), resulting in an optimistic bias in performance. In future trials, an "unseen" dataset can be introduced to calibrate the model.

The conclusions of this work are currently limited to small LLMs, and whether the observed reasoning-preservation benefits generalize to larger models remains an open question requiring further validation. Furthermore, our evaluation primarily focuses on reasoning performance and does not comprehensively assess potential cross-capability degradation in general knowledge, factuality, language understanding, or safety, which should be investigated in future work.

6 Reproducibility

To ensure complete transparency and enable exact replication of our empirical evaluations, all codebase components, configuration manifests, dataset splits, and evaluation scripts are included in the supplementary materials accompanying this submission.

7 AI Usage

In this work, we used generative AI tools for designing research methodologies, refining hypotheses, and providing feedback on experimental design. We have not used generative AI tools for generating synthetic datasets or formulating mathematical proofs, and other required disclosure subtasks are not

applicable to this work. Additionally, generative AI tools were used to search for and summarize relevant literature and edit the manuscript to improve readability and grammatical precision.

8 CONCLUSION

In this paper, we propose LLM-QRP, a layer-aware quantization framework for preserving reasoning performance in small LLMs for edge deployment. By combining SLED (Self-Logit Evolution Decoding) with structural entropy sensitivity (ΔH), we identify layers that play a critical role in the model's reasoning process and use these signals to guide mixed-precision quantization. Our ablation studies further validate that high-scoring layers are more important to preserve than low-scoring or randomly selected layers. Across multiple model architectures and benchmarks, LLM-QRP achieves a favourable accuracy compression Pareto frontier, substantially reducing model size and VRAM usage while maintaining near-baseline reasoning performance. These results demonstrate that selectively preserving reasoning-critical layers can provide a practical alternative to uniform quantization, enabling more efficient deployment of small LLMs on resource-constrained edge devices.

REFERENCES

URL https://ollama.com/huihui_ai/qwen3.5-abliterated:0.8B-fp16.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025. URL https://arxiv.org/abs/2502.02737.

Alexander Amini, Anna Banaszak, Harold Benoit, Arthur Böök, Tarek Dakhran, Song Duong, Alfred Eng, Fernando Fernandes, Marc Härkönen, Anne Harrington, Ramin Hasani, Saniya Karwa, Yuri Khrustalev, Maxime Labonne, Mathias Lechner, Valentine Lechner, Simon Lee, Zetian Li, Noel Loo, Jacob Marks, Edoardo Mosca, Samuel J. Paech, Paul Pak, Rom N. Parnichkun, Alex Quach, Ryan Rogers, Daniela Rus, Nayan Saxena, Bettina Schlager, Tim Seyde, Jimmy T. H. Smith, Aditya Tadimeti, and Neehal Tumma. Lfm2 technical report, 2025. URL https://arxiv.org/abs/2511.23404.

AutoGPTQ. Autogptq/autogptq: An easy-to-use llms quantization package with user-friendly apis, based on gptq algorithm. URL https://github.com/AutoGPTQ/AutoGPTQ.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023. URL https://arxiv.org/abs/2309.16609.

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. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale, 2022. URL https://arxiv.org/abs/2208.07339.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023a. URL https://arxiv.org/abs/2305.14314.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. Spqr: A sparse-quantized representation for near-lossless llm weight compression, 2023b. URL https://arxiv.org/abs/2306.03078.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization, 2024. URL https://arxiv.org/abs/2401.06118.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference, 2021. URL https://arxiv.org/abs/2103.13630.

Simla Burcu Harma, Danila Mishin, Zhengyuan Su, Ayan Chakraborty, Elizaveta Kostenok, Dongho Ha, Babak Falsafi, Martin Jaggi, Yunho Oh, and Amir Yazdanbakhsh. Mxsens: Sensitivity-aware mixed-precision quantization for efficient llm inference, 2026. URL https://arxiv.org/abs/2607.17733.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Xinyi Hou, Yanjie Zhao, and Haoyu Wang. Llm applications: Current paradigms and the next frontier, 2025. URL https://arxiv.org/abs/2503.04596.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Qinshuo Liu, Xianglong Liu, Luca Benini, Michele Magno, Shiming Zhang, and Xiaojuan Qi. Slim-llm: Salience-driven mixed-precision quantization for large language models, 2025. URL https://arxiv.org/abs/2405.14917.

IBM Research. Granite 4.2 language models. https://huggingface.co/blog/ibm-granite/granite-4-2, 2026. Accessed: 2026-07-25.

Ferdinand Kapl, Emmanouil Angelis, Tobias Höppe, Kaitlin Maile, Johannes von Oswald, Nino Scherrer, and Stefan Bauer. Do depth-grown models overcome the curse of depth? an in-depth analysis, 2025. URL https://arxiv.org/abs/2512.08819.

Sangjun Lee, Seung taek Woo, Jungyu Jin, Changhun Lee, and Eunhyeok Park. Amq: Enabling automl for mixed-precision weight-only quantization of large language models, 2025. URL https://arxiv.org/abs/2509.12019.

Zhen Li, Yupeng Su, Songmiao Wang, Runming Yang, Congkai Xie, Aofan Liu, Ming Li, Jiannong Cao, Yuan Xie, Ngai Wong, and Hongxia Yang. Quantization meets reasoning: Exploring and mitigating degradation of low-bit llms in mathematical reasoning, 2026. URL https://arxiv.org/abs/2505.11574.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration, 2026. URL https://arxiv.org/abs/2306.00978.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2022. URL https://arxiv.org/abs/2109.07958.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization, 2021. URL https://arxiv.org/abs/2106.08295.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Zhenyu Wang. Logitlens4llms: Extending logit lens analysis to modern large language models, 2025. URL https://arxiv.org/abs/2503.11667.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models, 2024. URL https://arxiv.org/abs/2211.10438.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers, 2022. URL https://arxiv.org/abs/2206.01861.

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, Will Song, Tiffany Zhao, Pranav Raja, Charlotte Zhuang, Dylan Slack, Qin Lyu, Sean Hendryx, Russell Kaplan, Michele Lunati, and Summer Yue. A careful examination of large language model performance on grade school arithmetic, 2024. URL https://arxiv.org/abs/2405.00332.

Jianyi Zhang, Da-Cheng Juan, Cyrus Rashtchian, Chun-Sung Ferng, Heinrich Jiang, and Yiran Chen. Sled: Self logits evolution decoding for improving factuality in large language models, 2025. URL https://arxiv.org/abs/2411.02433.

Xinghao Zhao. Entropy trajectory shape predicts llm reasoning reliability: A diagnostic study of uncertainty dynamics in chain-of-thought, 2026. URL https://arxiv.org/abs/2603.18940.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate llm serving, 2024. URL https://arxiv.org/abs/2310.19102.

A LLM-QRP ALGORITHM

The complete LLM-QRP sub-component profiling and 0-1 Knapsack bit-allocation procedure is presented in Algorithm 1.

Algorithm 1: LLM-QRP: Quantization via Reasoning Prioritization

Input : Model $\mathcal{M}$ with N layers, calibration set $\mathcal{P}$, reasoning token mask $\mathcal{T}_{\text{CoT}}$, target bit-width B_{target}, candidate bit-widths $\mathcal{B} = \{3, 4, 6, 8, 16\}$

Output : Quantized model $\mathcal{M}^*$ and optimal bit assignment matrix $\mathbf{x}^*$

for *each prompt* $p_k \in \mathcal{P}$ **do**
  Extract activation states over reasoning tokens $t \in \mathcal{T}_{\text{CoT}}$;
  for *each layer* $l = 1, \ldots, N$ **do**
    for *each sub-component* $c \in \{Q, K, V, O, \mathit{Gate}, \mathit{Up}, \mathit{Down}\}$ **do**
      $S^{(k)}_{\text{SLED}}(l, c) \leftarrow \text{SLED}(W_{l,c})\Big|_{t \in \mathcal{T}_{\text{CoT}}}$;
      $\Delta H^{(k)}(l, c) \leftarrow |H(P_{l,c}) - H(P_{l-1})|\Big|_{t \in \mathcal{T}_{\text{CoT}}}$;

$\mathbf{S}_{l,c} \leftarrow \frac{1}{|\mathcal{P}|} \sum_k S^{(k)}_{\text{SLED}}(l, c) \quad \forall (l, c)$;

$\boldsymbol{\Delta H}_{l,c} \leftarrow \frac{1}{|\mathcal{P}|} \sum_k \Delta H^{(k)}(l, c) \quad \forall (l, c)$;

Form feature matrix $\mathbf{X} \in \mathbb{R}^{(7N) \times 2}$ from pairs $(\mathbf{S}_{l,c}, \boldsymbol{\Delta H}_{l,c})$;

Fit PCA on $\mathbf{X}$ and extract principal component vector $\mathbf{w} = [w_1, w_2]^T$;

$R^{\text{raw}}_{l,c} \leftarrow w_1 \cdot \mathbf{S}_{l,c} + w_2 \cdot \boldsymbol{\Delta H}_{l,c} \quad \forall (l, c)$;

$R_{l,c} \leftarrow \text{MINMAXSCALE}\left(R^{\text{raw}}_{l,c}\right) \quad \forall (l, c)$;

Formulate bit allocation ILP:

$$\mathbf{x}^* = \underset{x_{l,c,b} \in \{0,1\}}{\arg\max} \sum_{l=1}^{N} \sum_{c} \sum_{b \in \mathcal{B}} x_{l,c,b} \cdot R_{l,c} \cdot b$$

$$\text{s.t.} \quad \sum_{l=1}^{N} \sum_{c} \sum_{b \in \mathcal{B}} x_{l,c,b} \cdot |W_{l,c}| \cdot b \leq B_{\text{target}} \cdot \sum_{l,c} |W_{l,c}|$$

$$\sum_{b \in \mathcal{B}} x_{l,c,b} = 1 \quad \forall (l, c)$$

;

Solve ILP via branch-and-bound to obtain binary indicator tensor $\mathbf{x}^*$;

$\mathcal{M}^* \leftarrow \text{QUANTIZESUBBLOCKS}\left(\mathcal{M}, \mathbf{x}^*\right)$;

Evaluate $\mathcal{M}^*$ on downstream benchmarks $\mathcal{D}$ and return results $\mathcal{R}$;

B Additional Quantization Experimental Details

Table 2: Full primary comparison across all four models (Appendix). GSM8K, TruthfulQA, and MMLU retained probability $\exp(-\text{NLL})$ (5-seed means), average effective bits-per-weight, and on-disk footprint.

Model	Method	B_{avg} (bits)	Footprint (MB)	GSM8K	TruthfulQA	MMLU
SmolLM2-135M	BF16	16.0	212.3	0.2005	0.0455	0.0080
	Uniform INT8	8.0	106.2	0.1960	0.0448	0.0082
	Uniform INT4	4.0	53.1	0.1528	0.0359	0.0051
	GPTQ (4-bit)	4.0	53.1	0.1471	0.0260	0.0038
	AWQ (4-bit)	4.0	53.1	0.1784	0.0412	0.0065
	SpQR (3-bit)	3.0	55.7	0.1834	0.0430	**0.0082**
	SliM-LLM (2-bit)	2.0	26.5	0.0007	0.0000	0.0000
	SmoothQuant (8-bit)	8.0	106.2	0.1991	**0.0452**	0.0080
	Atom (4-bit)	4.0	53.1	0.1199	0.0298	0.0042
	LLM-QRP	7.76	103.0	**0.1996**	0.0439	0.0080
LFM2-350M	BF16	16.0	392.2	0.4001	0.0393	0.0057
	Uniform INT8	8.0	196.1	**0.4018**	0.0394	0.0056
	Uniform INT4	4.0	98.0	0.3750	0.0376	0.0055
	GPTQ (4-bit)	4.0	98.0	0.3862	0.0320	0.0043
	AWQ (4-bit)	4.0	98.0	0.3767	0.0355	0.0049
	SpQR (3-bit)	3.0	102.6	0.3984	0.0365	0.0050
	SliM-LLM (2-bit)	2.0	49.0	0.0625	0.0003	0.0001
	SmoothQuant (8-bit)	8.0	196.1	0.3999	0.0391	**0.0057**
	Atom (4-bit)	4.0	98.0	0.3547	0.0349	0.0051
	LLM-QRP	5.96	146.1	0.3927	**0.0395**	0.0052
Qwen2.5-0.5B	BF16	16.0	715.7	0.4938	0.0340	0.0065
	Uniform INT8	8.0	357.8	0.4845	0.0343	0.0059
	Uniform INT4	4.0	178.9	0.4022	**0.0354**	**0.0085**
	GPTQ (4-bit)	4.0	178.9	0.3948	0.0239	0.0045
	AWQ (4-bit)	4.0	178.9	0.4531	0.0350	0.0072
	SpQR (3-bit)	3.0	188.5	0.4855	0.0338	0.0062
	SliM-LLM (2-bit)	2.0	89.5	0.0307	0.0001	0.0001
	SmoothQuant (8-bit)	8.0	357.8	0.4933	0.0341	0.0065
	Atom (4-bit)	4.0	178.9	0.3873	0.0278	0.0059
	LLM-QRP	7.88	352.6	**0.4939**	0.0333	0.0064
Granite-4.0-350M	BF16	16.0	499.1	0.2521	0.0378	0.0077
	Uniform INT8	8.0	249.6	0.0939	0.0051	0.0004
	Uniform INT4	4.0	124.8	0.1283	0.0283	0.0060
	GPTQ (4-bit)	4.0	124.8	0.2125	0.0264	0.0043
	AWQ (4-bit)	4.0	124.8	0.1934	0.0359	0.0059
	SpQR (3-bit)	3.0	130.7	0.2363	**0.0382**	**0.0106**
	SliM-LLM (2-bit)	2.0	62.4	0.0096	0.0001	0.0001
	SmoothQuant (8-bit)	8.0	249.6	**0.2464**	0.0380	0.0075
	Atom (4-bit)	4.0	124.8	0.1474	0.0289	0.0045
	LLM-QRP	7.72	240.8	0.2462	0.0376	0.0075

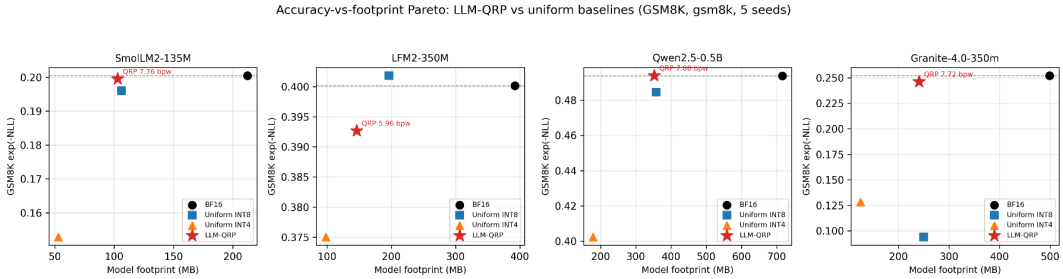

Figure 3: Trade-off analysis between model compression ratio and accuracy across target benchmarks.

B.1 Retention Ablation Tests

Table 3: Retention ablation: mean, standard deviation, and percentage of BF16 retained after quantizing the least- or most-active 20% of sub-components to 4-bit, ranked by each candidate signal, plus a shared random, uniform, and BF16 reference.

Model	Rank signal	Rule	Mean	Std	Retained (%)
SmolLM2-135M	ΔH	bottom	0.1930	0.0038	96.3
		top	0.1868	0.0039	93.2
	R (PCA)	bottom	0.1905	0.0030	95.1
		top	0.1883	0.0048	94.0
	SLED	bottom	0.1905	0.0030	95.1
		top	0.1880	0.0046	93.8
	–	random	0.1930	0.0075	96.3
		uniform	0.1541	0.0034	76.9
		bf16	0.2004	0.0045	100.0
Qwen2.5-0.5B	ΔH	bottom	0.4825	0.0013	98.4
		top	0.4818	0.0037	98.2
	R (PCA)	bottom	0.4825	0.0013	98.4
		top	0.4818	0.0037	98.2
	SLED	bottom	0.4827	0.0037	98.4
		top	0.4830	0.0017	98.5
	–	random	0.4823	0.0096	98.4
		uniform	0.4046	0.0017	82.5
		bf16	0.4904	0.0029	100.0