CERTIFIED BEST-FIRST SPARSE ATTENTION: PROVABLE EARLY TERMINATION VIA ADMISSIBLE BLOCK BOUNDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse attention makes long-context decoding affordable, but most methods silently discard KV blocks: they do not report how much attention mass was dropped. Fixed-budget selectors commit to a budget independently of the query, dynamic heuristics adapt but provide no bound, and probabilistic verifiers offer only high-probability guarantees at a fixed configuration budget. We recast sparse attention as best-first search with provable early termination. BNB-Attention reads KV blocks in decreasing order of an admissible upper bound on the attention scores inside each block, and stops once the unread blocks can contain at most an ε fraction of the attention mass accumulated so far. This yields a deterministic, per-query, per-head certificate bounding the output error relative to dense attention; in the worst case, the method falls back to dense attention. The practicality of the certificate hinges on bound tightness: on real activations, min/max and ball bounds are 22–24 nats loose, forcing near-dense visits, whereas per-block quantized bounds with explicit rounding radii are roughly $80\times$ tighter, making certified early termination viable. Across more than 19 million evaluations on seven checkpoints from four base model families, we observe zero certificate violations. At $\varepsilon = 10^{-2}$, BNB-Attention matches dense performance on RULER32K-HARD using only 3.1% KV density, remains within sampling noise of dense on AIME 2024 where certificate-free SnapKV collapses, and matches the strongest probabilistic verifier on its own LongBench protocol at roughly one-third the KV budget. It also achieves $\sim 2.7\times$ attention-kernel speedup over FlashAttention at 128K context under the quality-verified setting $\varepsilon = 10^{-2}$.

1 INTRODUCTION

Long-context decoding pays softmax attention's quadratic cost, and training-free sparse attention is the usual way to avoid it. The shortcut is rarely audited. *Fixed-budget* selectors — top-k by proxy score, eviction policies such as H2O [1], SnapKV [2], Quest [4] — read a fixed number of blocks per query, ranked by a proxy score that uses the query but is not certified, so the budget is committed regardless of how much attention mass those blocks actually carry. *Dynamic* heuristics adapt to the query, but nothing bounds what they silently drop; the error surfaces downstream, if at all. That risk is real: learned routers co-adapt until they perform no better than random gates [18], and sparse variants of frontier models ship with accuracy monitored only downstream.

We ask: *can a training-free sparse attention method certify, per query and per attention head, that its attention output is ε-close to dense attention — and stop computing the moment that certificate holds?*

Search, not selection. Our answer is BNB-ATTENTION (best-first branch and bound). Each block of keys gets an upper bound U_b on every attention score inside it, and the blocks are read in decreasing order of that bound, causal diagonal first (Figure 1). After each block, we test whether the unread blocks can still hold more than ε times the attention mass accumulated so far; when they cannot, the query is certified and the search stops. Peaked queries stop after a few blocks, diffuse queries pay for more context, and the worst case is exactly dense attention. We prove the stopping rule sound, the certificate tight, and the best-first order optimal among all orders that use the same bounds.

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.

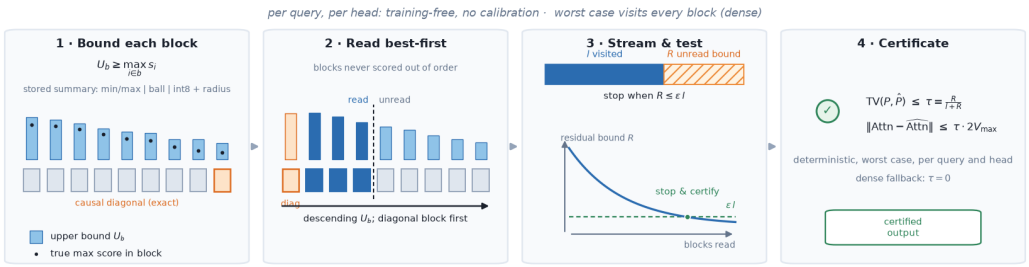

Figure 1: Overview of BNB-ATTENTION, per query and head: (1) every block stores a cheap admissible bound on its scores; (2) blocks are read best-first, causal diagonal first; (3) the visited mass l accumulates while the unread blocks contribute at most the bound R; (4) once $R \leq \varepsilon l$ the search stops and the output is certified, with dense attention as the worst case.

Adaptivity is necessary; the rest is verified. No fixed budget can certify in principle: for any budget $k < (1-\varepsilon)|\mathcal{P}|$ there are activations on which *every* k-block selector incurs truncation error $1 - k/|\mathcal{P}| > \varepsilon$ — whichever blocks it picks and whichever summaries it computes — while BNB-ATTENTION certifies every query (Theorem 3). The design choices that remain are verified rather than assumed: the certificate is tight (Proposition 3), each bound family is the pointwise-smallest admissible function of its summary (Proposition 4), and quantized bounds are 2^b tighter than axis-aligned ones under a Gaussian block model (Propositions 5–6). These are consistency results, not deep lower bounds; a necessity result for quantized bounds remains open.

Bound tightness is the bottleneck. The certificate's strength is governed by the looseness $U_0 - m$ of the best block, and the three families differ sharply. Quest-style min/max bounds and centroid+radius bounds are 22.3–24.2 nats loose — so loose that certified termination visits nearly everything. Per-block quantized keys with an explicit rounding radius (RaBitQ-style [20]) cut looseness to 0.29 nats, $\sim 100\times$ tighter, and turn certified early termination into a practical regime. At decode, where scoring is a cheap GEMV, BNB-ATTENTION can score exactly, and the certificate becomes the *exact* tail mass: τ is the true total-variation distance of the truncation.

Contributions.

- **Framework and theory.** Sparse attention as best-first search under admissible bounds, with provable early termination and deterministic per-query, per-head error certificates (Section 3; Propositions 3–4).
- **The bound hierarchy.** Quantized bounds, not min/max or ball, make certification practical: 77–83$\times$ tighter in nats, a $\sim 10^9\times$ larger certifiable region, explained by the 2^b rounding-radius scaling (Table 1; Corollary 1).
- **The certificate isolated, then validated.** At matched budget, certified stopping beats the uncertified heuristic 1.4–1.7$\times$ and matches an oracle with exact scores on the median; zero violations across 19,046,400 evaluations; perplexity equal to dense to three decimals.
- **Deployment evidence.** RULER32K-HARD matched dense at 3.1% KV density on two model families; AIME within one standard error while certificate-free SnapKV scores 0/30; parity with the strongest probabilistic verifier on its own LongBench protocol at a third of its KV budget; $\sim 2.7\times$ FlashAttention at 128K and 1.20$\times$ end-to-end (Sections 4.1–4.4).

Section 3 develops the method; Sections 4.1–4.4 evaluate it.

2 RELATED WORK

Training-free sparse attention. H2O [1] evicts low-score keys; SnapKV [2] selects keys from an observation window; PyramidKV [3] allocates budgets across layers; Quest [4] uses per-page

min/max bounds as a top-k proxy; MagicPIG [5] hashes keys with LSH; MInference [7] profiles per-head patterns offline. Recent entries add binary-PC hashing [21], rotated binary quantization for adaptive top-p [8], proxy–kernel co-design [9], RoPE frequency sparsity [23], and cross-layer index reuse [10, 11]. All of them fix a selection rule and offer no output guarantee. Native sparse attention (NSA, DSA) [12, 13] is trained, unlike ours. Post-hoc theory bounds the truncation error only after the logits are fixed [14, 15, 16], and none of it decides *when to stop reading keys*; sampling approaches control error in expectation, not per query.

Probabilistic verification: VATTENTION [6]. The closest system combines sink/local/top-k selection with uniform random sampling, so that its estimates of numerator and denominator are (ϵ, δ)-accurate with probability $1-\delta$. Three structural differences separate it from BNB-ATTENTION. *Guarantee class:* vAttention bounds error on *estimates* of the numerator and denominator, in probability, and draws a base sample before every budget decision; our certificate is an exact worst-case bound on tail mass given admissible summaries. *Budget semantics:* vAttention's budget is a per-configuration target (base rate and top-k, with per-row sizing for its (ϵ, δ) bound), whereas ours adapts per query — from a single block on concentrated queries to full context where certification demands it. *Composition:* vAttention's accuracy depends on the top-k predictor it wraps (oracle vs. HashAttention spans several points), while admissible bounds keep our certificate valid for any bound family, reducing efficiency to bound tightness, which we quantify. Their CPU-offloaded KV system is complementary to our kernel-level results. To our knowledge, no prior training-free method produces an *a-priori*, deterministic, per-query, online-stopped certificate of this kind while also supporting exact, drop-in decoding at long context.

3 METHOD

Setup. For one query $q \in \mathbb{R}^D$ (scale $\sigma = 1/\sqrt{D}$) and a causal KV cache $\{(k_i, v_i)\}_{i \le p}$, partition the keys into blocks of size B. Write $s_i = \sigma\, q^\top k_i$ for the score of token i, $Z = \sum_{i \le p} e^{s_i}$, and $\mathrm{Attn}(q) = \sum_i (e^{s_i}/Z)\, v_i$. The diagonal block $d = \lfloor p/B \rfloor$ is computed exactly under the causal mask; the pool blocks $\mathcal{P} = \{b < d\}$ are the ones we search. The search must decide which pool blocks matter without scoring their tokens first: that is what the bounds are for.

Admissible bounds. A per-block function $U_b(q)$ is *admissible* if $U_b(q) \ge \max_{i \in b} s_i$: it provably upper-bounds every score in the block, so a block's contribution can be bounded before any of its tokens are read. We use three families, each stored as a compact per-block summary:

- **min/max** (Quest-style): with per-dimension block extrema $k^{\min}, k^{\max}$, the bound is $U_b = \sigma \sum_d \max(q_d k_d^{\max},\, q_d k_d^{\min})$;
- **ball**: $U_b = \sigma(q^\top c_b + \|q\| R_b)$, from the block centroid c_b and radius R_b;
- **quant**: per-block, per-dimension symmetric INT8 codes $k_i = \hat{k}_i + e_i$ with $|e_{i,d}| \le r_d$, giving $U_b = \sigma(\max_{i \in b} q^\top \hat{k}_i + \sum_d |q_d| r_d)$.

All three are admissible by construction (Theorem 4, Appendix H). They differ in what they store and how tight they are: min/max and ball are cheap and loose, while quant stores compressed keys and is far tighter (Section 4.1).

Best-first search with early termination. Algorithm 1 reads the pool blocks in decreasing order of U_b under online softmax [19], maintaining (m, l, acc). With $U_0 = \max_b U_b$ and shifted block masses $w_b = |b| \exp(U_b - U_0)$, it precomputes the suffix masses $S_j = \sum_{b \ge j} w_b$ in visit order. After rank j, the unread blocks can hold at most $\mathrm{residual}_j = S_{j+1} e^{U_0 - m}$ of shifted mass, so the search stops once $\mathrm{residual}_j \le \varepsilon\, l$: at that point the tokens left unread weigh at most a fraction ε of those already read.

3.1 THEORY

Let $\hat{P}$ denote the softmax over visited tokens (renormalized) and P the full softmax; let $V_{\max} = \max_i \|v_i\|$. Proofs are in Appendix H.

Algorithm 1 BNB-ATTENTION: certified best-first sparse attention (per query, per head)

Require: q, KV blocks, admissible bounds U, tolerance ε

compute diagonal output stats (m, l, acc) over block d (causal mask)
order pool $\mathcal{P}$ by descending U_b with $U_0 = \max_b U_b$; suffix masses $S_j = \sum_{b \geq j} |b| e^{U_b - U_0}$ in visit order
for rank $j = 0, 1, \ldots, |\mathcal{P}| - 1$ **do**
  visit block b_j: online-softmax update of (m, l, acc)
  if $S_{j+1}\, e^{U_0 - m} \leq \varepsilon\, l$ **then**
    return acc/l with certificate $\tau = \frac{S_{j+1} e^{U_0 - m}}{l + S_{j+1} e^{U_0 - m}}$
  end if
end for
return dense output (residual 0, trivially certified with $\tau = 0$)

Theorem 1 (Certificate)**.** *At any point of Algorithm 1 with running statistics (m, l), let $R = S_{j+1}\, e^{U_0 - m}$ be the residual bound over the unvisited pool. Then*

$$\mathrm{TV}(P, \hat{P}) \;\leq\; \tau := \frac{R}{l + R}, \qquad \left\| \mathrm{Attn}(q) - \widehat{\mathrm{Attn}}(q) \right\| \;\leq\; \tau \cdot 2V_{\max}.$$

The proof is two lines. Admissibility gives $Z_T e^{-m} \leq R$ for the discarded mass, and $x/(l + x)$ is monotone, so $\mathrm{TV} = Z_T e^{-m}/(l + Z_T e^{-m}) \leq R/(l + R)$. The output bound follows from the head–tail decomposition, with both means inside the radius-$V_{\max}$ ball. In words: τ is the largest fraction of the attention mass that could have been dropped, and the output can move by at most τ times the diameter of the value cloud, $2V_{\max}$.

Theorem 2 (Termination; graceful degradation)**.** *Algorithm 1 terminates for every query within $|\mathcal{P}|+1$ visited blocks; if the pool exhausts, the output* equals *dense attention and the certificate holds with equality. In finite precision the check can fail by an accumulation-order margin; we audit in fp64 and give a floating-point-safe guard (Proposition 7).*

Theorem 3 (No fixed budget can certify)**.** *Fix $\varepsilon \in (0, 1)$ and n_B equal-size blocks. For every budget $k < (1 - \varepsilon) n_B$ there is a query–cache configuration on which* every *selector visiting exactly k pool blocks incurs $\mathrm{TV}(P, \hat{P}) = 1 - k/n_B > \varepsilon$ — whichever blocks it picks, whichever summaries it computes. On the same configuration class, Algorithm 1 certifies every query.*

The remaining design choices are verified, not assumed, each in one line. *Order optimality*: descending U_b minimizes the residual bound at every visited count among all orders that use the same bounds, and at decode — where exact scores cost one GEMV — the certificate becomes the *exact* TV distance (Theorem 1; Proposition 2). *Tightness*: there are activations where $\mathrm{TV} = \tau$ exactly, so Theorem 1 cannot be sharpened without changing the bounds (Proposition 3). *Pointwise optimality*: among all functions of a block summary, min/max, ball, and quantized bounds are the pointwise-smallest admissible ones, so our hierarchy compares optimal representatives (Proposition 4). *The 2^b law*: quantized bounds pay a rounding radius $2^b - 1$ smaller than either axis-aligned family (Proposition 5); under a Gaussian block model, $\ell_{\mathrm{quant}} = \Theta(\sigma_k \sqrt{\log B}/2^b)$ versus $\Theta(\sigma_k \sqrt{\log B})$ (Proposition 6). The consequence is a *looseness barrier*: termination requires $\tau^* \leq \varepsilon e^{-\lambda_*}$ with λ_* the smallest unvisited-block looseness, so a Δ-nat reduction in looseness multiplies the certifiable region by e^{Δ} (Corollary 1). What remains open: a necessity result for quantized bounds, and expected visited blocks under realistic score models.

Proposition 1 (Order optimality among fixed bounds)**.** *Fix admissible bounds $\{U_b\}$ with equal-size pool blocks. Among all block orders of $\mathcal{P}$, descending U_b minimizes the residual bound $\sum_{b \text{ unvisited}} |b| e^{U_b - m}$ at every visited count j and running max m.*

Proposition 2 (Exact certificates at decode)**.** *When exact scores of all tokens are available (decode: one GEMV), ordering by exact block maxima with the exact suffix mass makes τ equal to the true $\mathrm{TV}(P, \hat{P})$.*

Proposition 3 (Tightness of the certificate)**.** *There are activations on which Algorithm 1 stops with $\mathrm{TV}(P, \hat{P}) = \tau$ exactly: Theorem 1 cannot be sharpened without changing the bounds.*

bound family	storage/block	looseness $U_0 - m$ (nats)	sparsity tracks ε?	certifies early?
min/max	$2D$	22.3	no	no
ball	$D{+}1$	24.2	no	no
quant8	BD int8 $+ D$	0.29	yes	yes

Table 1: The bound hierarchy on real activations (mean over layers 6/12/23, all heads, 2048 queries each). Looseness at the argmax block enters the residual as the multiplicative factor $e^{U_0 - m}$.

Proposition 4 (Pointwise optimality within a summary class)**.** *Among all functions of the per-dimension extrema $(k^{\min}, k^{\max})$, the min/max bound is the pointwise-smallest admissible function; among all functions of (c_b, R_b), the ball bound is the pointwise-smallest admissible function; among all functions of the quantized codes $(\hat{k}_i, r_d)_{i \in b}$, the quant bound is the pointwise-smallest admissible function.*

Proposition 5 (2^b domination)**.** *Let $\rho_d = (k_d^{\max} - k_d^{\min})/2$ and $W = \sigma \sum_d |q_d| \rho_d$. For b-bit symmetric range quantization, $\ell_{\text{quant}} \le \frac{2}{2^b - 1} W$ vs. $\ell_{\min\,/\,\max} \le 2W$: tighter by $2^b - 1$.*

Proposition 6 (Gaussian block model)**.** *If keys in a block are i.i.d. $\mathcal{N}(0, \sigma_k^2)$ per dimension and $q \sim \mathcal{N}(0, I/D)$, then with probability $1 - o(1)$, $\ell_{\text{quant}} = \Theta(\sigma_k \sqrt{\log B}/2^b)$ vs. $\ell_{\min\,/\,\max} = \Theta(\sigma_k \sqrt{\log B})$: tighter by $\Theta(2^b)$.*

Corollary 1 (Looseness barrier)**.** *If the certificate passes, the true tail fraction obeys $\tau^* \le \varepsilon e^{-\lambda_*}$ with λ_* the smallest unvisited-block looseness; a Δ-nat reduction in looseness multiplies the certifiable region by e^{Δ}. With the measured gap,* $\exp(22.3 - 0.29) \approx 3{\times}10^9$.

Proposition 7 (Floating-point-safe stopping)**.** *With statistics evaluated in working precision u, the guarded rule* $\text{residual}_j \le (\varepsilon - \delta_{\text{fp}}) l$ *with $\delta_{\text{fp}} = c' u (n_{\text{vis}} B + n_B)$ preserves $\text{TV}(P, \hat{P}) \le \varepsilon$ in floating point; at fp32 the guard predicts excesses of order 10^{-5}–10^{-3}, bracketing the 2.9×10^{-4} maximum observed (Section 4.1).*

Complexity and memory. The bound pass reads compressed keys once ($O(n_B BD)$; INT8 at half the fp16 footprint), and the exact stage re-reads fp16 K and V for visited tokens. With visited token fraction ρ, traffic is $(1{+}2\rho)/2$ of dense (ceiling $2/(1{+}2\rho)$; $4/(1{+}4\rho)$ with INT8-K). The int8 index costs $\approx 1.01LD$ bytes — half the fp16 K it replaces, $+25\%$ of the KV both we and the baselines keep; Quest-class page statistics are far smaller ($\approx 0.016LD$ at $B{=}128$), so the index is the price of auditable low density. An int8-only variant removes the fp16 K entirely (Appendix M).

4 Experiments

Two testbeds. A local single GPU hosts the controlled probes (Qwen2.5-0.5B, block $B{=}128$, FineWeb-Edu; the only knob is ε). A high-memory accelerator hosts the deployment-scale models (five 7–8B checkpoints across four families, fp16, the same code unchanged). All certificates are verified in exact fp64 arithmetic.

4.1 Certificates, bound tightness, and the certificate-off control

Zero violations. Across 19,046,400 query-head/position/ε evaluations on seven checkpoints, the measured error never exceeded the certified bound (0 violations), and certification succeeded on 100% of queries at every ε (Theorem 2). Realized error averages below 1% of the certified envelope: the bound is a worst case that is almost never approached, which also explains why the fp32 accumulation-order excesses (678 instances, 0.04%, max $2.9{\times}10^{-4}$, all final-layer) vanish under fp64 verification (Appendix K).

The bound hierarchy. Table 1 is the central empirical finding. Min/max and ball bounds are 22.3–24.2 nats loose and cannot certify before visiting almost everything; quantized bounds are 0.29 nats — 77–83$\times$ tighter, a $\sim 10^9\times$ smaller multiplicative residual — and deliver sparsity that tracks ε (up to 72.0% block sparsity at $\varepsilon{=}0.1$ on mid layers; all quality results use $\varepsilon \le 10^{-2}$). The measured separation is smaller than the Gaussian model's $2^8{=}256\times$ because real blocks are asymmetric: the model predicts the direction and order of magnitude, not the constant (Appendix K).

model / regime	stopping rule	mean rel. err	median	blocks
Llama-3.1-8B-Instruct — LM @8K				
	BnB certified (ε=10^{-2})	1.5e-03	2.3e-04	34.2
	heuristic bound-mass stop	2.2e-03	5.8e-04	33.4
	oracle top-p (exact scores)	1.1e-04	2.9e-07	35.5
Llama-3.1-8B-Instruct — NIAH @32K				
	BnB certified (ε=10^{-2})	1.8e-03	7.2e-04	116.9
	heuristic bound-mass stop	2.6e-03	1.2e-03	113.7
	oracle top-p (exact scores)	2.4e-04	1.1e-04	123.8
R1-Distill-Llama-8B — LM @8K				
	BnB certified (ε=10^{-2})	1.6e-03	4.4e-04	34.0
	heuristic bound-mass stop	2.4e-03	9.3e-04	33.2
	oracle top-p (exact scores)	1.1e-04	3.0e-07	35.6
R1-Distill-Llama-8B — NIAH @32K				
	BnB certified (ε=10^{-2})	2.1e-03	1.1e-03	117.4
	heuristic bound-mass stop	3.0e-03	1.7e-03	114.3
	oracle top-p (exact scores)	2.3e-04	1.1e-04	125.5

Table 2: Certificate-on vs. certificate-off at matched visited-block budget: relative attention-output error vs. dense (fp32), 256 positions $\times$ all heads $\times$ four layers per cell; the heuristic and the oracle are calibrated so their mean visited-block count matches the certified rule's. The oracle receives exact scores and exact masses (unavailable without dense scoring); the heuristic is the certificate-off control.

Certificate-on vs. certificate-off, at matched budget. The natural objection: the wins come from adaptive stopping with tight bounds, and the certificate is a passenger. We test it directly, with the same quant8 bounds and best-first order, and three stopping rules: (i) certified; (ii) a heuristic bound-mass stop (no certificate); (iii) an oracle top-p with exact scores and masses. The heuristic and the oracle are calibrated to the certified rule's mean visited-block count, on 256 query positions (decode steps) $\times$ all heads $\times$ four layers from two checkpoints in two regimes (8K LM windows; 32K NIAH). Table 2: the certified rule beats the heuristic 1.4–$1.7\times$ on mean relative error (2–$5\times$ on the median) in every regime, and trails the oracle — handed the exact scores the bound pass exists to avoid computing — by $\sim$8–$10\times$. The certificate is not free accuracy; it is the price of not paying for exact scores, and unlike either alternative it is auditable at inference time.

4.2 Perplexity, frontier, and serving speed

Perplexity hides the damage. On $\sim$197k FineWeb-Edu tokens, BnB-Attention matches dense perplexity to three decimals (14.117) at ε=10^{-2} and 10^{-3} while skipping 55–62% of blocks. Matched-budget top-k and SnapKV do the same (within ±0.01): perplexity is too coarse to expose sparse-attention damage. On Qwen3-0.6B, matched top-k does degrade ($+0.012$–0.017; Appendix C). The evaluations that follow are the ones that discriminate.

The error–sparsity frontier. On the Pareto sweep (Appendix A), BnB-Attention traces the same trade-off as the strongest fixed selectors: 8–$50\times$ lower error than proxy top-k at equal sparsity, 100–$1000\times$ lower than random. It is also the only curve whose every point carries a per-query certificate. The proxy and true block-max top-k curves nearly coincide, which says that with tight quantized bounds the *ranking* is almost oracle-perfect; what fixed budgets cannot supply is the per-query *budget* itself.

Decode speed. The fused Triton pipeline (one score+stats kernel that reads K once, plus a split-K exact kernel) runs $\sim2.7\times$ FlashAttention at 128K at the quality-verified ε=10^{-2}, and up to $6.2\times$ at ε=10^{-1}, growing monotonically with context on all five 7–8B checkpoints (Figure 2). Perplexity agrees with dense to three decimals on every model (Appendix D). At 32K, BnB-Attention is also the sparsest by a wide margin: 3.1% decode density, against vAttention's 10% target and the $\approx$25% fixed budgets.

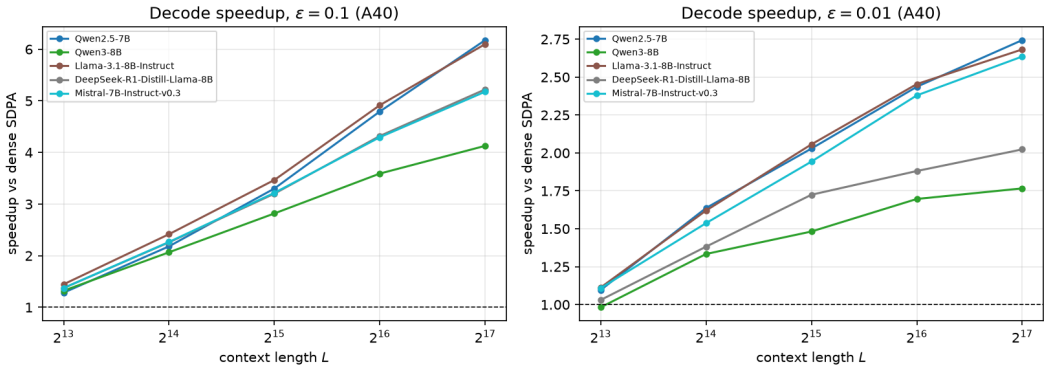

Figure 2: Decode speedup vs dense SDPA per model and ε (high-memory accelerator, CUDA graphs, best of the exact-score and INT8 pipelines).

End-to-end (full model). A full-model 32K-context generation of 128 tokens on DeepSeek-R1-Distill-Llama-8B — every layer, the MLP, and the bound pass — takes 35.9 s certified versus 42.9 s dense ($1.20\times$; the shared dense 32K prefill dominates both by design). The per-token decode delta is 55 ms, consistent with the $\sim 2.5\times$ kernel speedup at this operating point. Serving-stack integration (continuous batching, paged KV) is future work.

4.3 Probes that discriminate: retrieval and chain-of-thought faithfulness

Retrieval. A five-digit needle is planted at depths $\{0.1, \ldots, 0.9\}$ of a 32K haystack. Certified BNB-ATTENTION recovers it at every depth, matching dense exactly, while aggressive fixed budgets collapse (k=4: 20% hit; k=16: 40%; Appendix O). At matched budget, with VATTENTION calibrated per model, five depths $\times$ five seeds (25 cells per condition, Wilson CIs): every condition — certified, heuristic baselines, and VATTENTION at its calibrated rate — is statistically indistinguishable from dense, and VATTENTION spends 0.97–0.99 of its target budget (Table 31).

Chain-of-thought faithfulness. GSM8K problems are embedded after $\sim$8K distractor tokens; dense greedy CoT is rescored under each condition, and we report $\mathrm{KL}(P_{\text{dense}} \| P_{\text{cond}})$ and greedy argmax flips over CoT positions. At n=30 problems on two checkpoints (blocks calibrated to BNB-ATTENTION's budget per model): BNB-ATTENTION flips $\leq$0.2% of greedy next-token choices for $\varepsilon \leq 10^{-2}$ (0.5% at ε=10^{-1}), top-k flips 1.8–2.7%, and SnapKV is logit-faithful (0–0.7%) *yet collapses on RULER32K-HARD retrieval*. Low logit divergence and lost retrieval mass coexist — the silent-failure mode a per-query certificate is designed to rule out (Table 38).

4.4 Standard long-context benchmarks

We run four standard suites against the concurrent verified line, on three 8B checkpoints (Llama-3.1-8B-Instruct, DeepSeek-R1-Distill-Llama-8B, Mistral-7B-Instruct-v0.3) and up to five conditions (dense, certified BNB-ATTENTION, VATTENTION, top-k, SnapKV); failed cells are marked n/a, never imputed.

Standard LongBench at VATTENTION's own protocol. The cleanest comparison: VATTENTION's seven-task LongBench selection, its greedy protocol, its checkpoint, with the fixed baselines budget-matched to its 10% target (Table 3). BNB-ATTENTION at 3.1% density is the only condition that matches dense (mean 43.5 vs. 43.3); VATTENTION trails by 2.4 points, and budget-matched top-k/SnapKV lose 3.7/16.2. On VATTENTION's own evaluation and budget, the certificate is the cheaper and more faithful attribution.

RULER32K-HARD. Table 4: BNB-ATTENTION matches dense on both model families (75.8 vs. 74.7 on R1-Distill; 83.6 vs. 84.3 on Llama-3.1); VATTENTION tracks dense as well (74.8, 84.9);

Table 3: Standard LongBench on the seven-task selection of [6] (*their* model, protocol, and task list): Llama-3.1-8B-Instruct, greedy decoding, 128 new tokens, 50 samples per task. Fixed-budget baselines are capped to vATTENTION's own budget (k=8 $\Rightarrow$ 1024 tokens $\approx$ 10% of the KV cache at the mean context length; per-task density varies with L). BNB-ATTENTION at 3.1% density is the only condition matching dense; vATTENTION at its 10% target trails by 2.4 points, and the budget-matched fixed baselines lose 3.7–16.2.

task	Dense	BnB ε=10^{-2}	vATTENTION	top-k (k=8)	SnapKV (k=8)
multifieldqa-en	58.8	57.8	58.2	61.7	34.1
hotpotqa	53.3	53.3	48.9	44.8	47.1
narrativeqa	29.1	28.7	28.6	28.3	16.1
qasper	49.4	49.4	46.3	49.4	24.8
musique	31.6	34.1	28.0	17.6	20.3
qmsum	25.4	25.8	25.0	25.3	19.9
2wikimqa	55.7	55.7	51.0	50.5	27.3
Mean	**43.3**	**43.5**	40.9	39.7	27.1

Table 4: RULER32K-HARD, greedy decoding, 20 requests per subset, 32K context. Certified BNB-ATTENTION matches dense on both model families while certificate-free SnapKV collapses on multi-key retrieval; vATTENTION tracks dense on the same protocol. Budgets: BNB-ATTENTION ε=10^{-2} visits **3.1%** of the KV cache (decode density pooled over the benchmark; prompt-dependent, the timing harnesses measure 26–32% at 32K); vATTENTION targets 10%; top-k/SnapKV hold 64 blocks ($\approx$25% at 32K). VT = variable tracking, FWE = frequent word extraction, NIAH-MK = multi-key needle, NIAH-MV = multi-value needle.

task	Dense	BnB ε=10^{-2}	vATTENTION	top-k (k=64)	SnapKV (k=64)
DeepSeek-R1-Distill-Llama-8B					
QA-1	50.0	50.0	50.0	55.0	25.0
QA-2	55.0	65.0	55.0	55.0	20.0
VT	48.0	47.0	48.0	65.0	5.0
FWE	93.3	93.3	93.3	93.3	85.0
NIAH-MK2	100.0	100.0	100.0	95.0	0.0
NIAH-MK3	85.0	85.0	85.0	90.0	0.0
NIAH-MV	91.2	90.0	92.5	91.2	6.2
Mean	**74.7**	**75.8**	**74.8**	**77.8**	**20.2**
Llama-3.1-8B-Instruct					
QA-1	50.0	50.0	70.0	85.0	25.0
QA-2	50.0	50.0	40.0	50.0	15.0
VT	98.0	98.0	99.0	99.0	30.0
FWE	93.3	93.3	91.7	88.3	86.7
NIAH-MK2	100.0	95.0	95.0	100.0	0.0
NIAH-MK3	100.0	100.0	100.0	100.0	0.0
NIAH-MV	98.8	98.8	98.8	100.0	6.2
Mean	**84.3**	**83.6**	**84.9**	**88.9**	**23.3**

top-k posts the highest mean (77.8/88.9) — the denoising effect [6] also reports — but carries no error bound and collapses wherever its proxy ranking errs. Certificate-free SnapKV scores 0.0% on every multi-key retrieval subset. A 200-sample rerun reproduces every ordering (Appendix O).

LongBench-E and LooGLE-16K. On LongBench-E (300 samples per task, T=0.6), BNB-ATTENTION matches dense on every task (paired mean 34.9 vs. 34.3), and the passage-retrieval quartet is legible at a glance: **BnB** 90.2 > **dense** 87.9 > **top-k** 86.5 $\gg$ **SnapKV** 13.1. On LooGLE-16K every condition tracks dense (0.794–0.830 vs. 0.830/0.831) with SnapKV's long-dependency gap the largest (Appendix F).

AIME 2024. Table 5: on the one model with non-trivial accuracy at this scale (R1-Distill), BNB-ATTENTION, vATTENTION, and top-k land within sampling noise of dense (46.7%/53.3%/53.3% vs. 56.7%; n=30, $\pm$9.1 binomial). SnapKV collapses to 0/30 on *every* checkpoint — its

Table 5: AIME 2024 (30 problems, sampled decoding $T{=}0.6$; single-pass hub protocol unless marked Avg@4), 16K/4K generation caps. Budgets: BNB-ATTENTION $\varepsilon{=}10^{-2}$ visits **3.1%** of the KV cache during generation (measured); VATTENTION targets 10%; top-k/SnapKV hold $64{\times}128$ tokens. Certified BNB-ATTENTION matches dense within sampling noise on R1-Distill (the only model with strong reasoning accuracy); VATTENTION and top-k land in the same noise band. Certificate-free SnapKV collapses to $0/30$ on every checkpoint — its observation-window selection silently destroys long reasoning chains. Llama-3.1-8B-Instruct and Mistral-7B-v0.3 have near-floor single-pass accuracy at this scale, so conditions are only weakly discriminated there.

model	protocol	Dense	BnB $\varepsilon{=}10^{-2}$	VATTENTION	top-k	SnapKV
R1-Distill-Llama-8B	Avg@4 (legacy)	50.0%	47.5%	–	–	–
R1-Distill-Llama-8B	1× (hub)	56.7%	46.7%	53.3%	53.3%	0.0%
Mistral-7B-v0.3	1× (hub)	0.0%	3.3%	0.0%	0.0%	0.0%
Llama-3.1-8B-Inst	1× (hub)	3.3%	6.7%	13.3%	10.0%	0.0%

observation-window selection silently destroys long reasoning chains. Certified sparsity thus preserves dense behavior on the regime where silent trajectory drift is most dangerous, and says so at inference time.

5 DISCUSSION AND LIMITATIONS

What the certificate is for. Sparse-attention failures are silent: perplexity looks fine while retrieval-critical queries lose mass. A per-query τ makes the accuracy–compute trade-off auditable at inference time: tighten ε for reasoning tokens without retraining. The control experiment shows what the certificate buys (Table 2): matched for budget, it adds no accuracy over the uncertified heuristic and none over the oracle; what it adds is the *guarantee*, at the cost of a small gap to an oracle with exact scores. The bound pass reads the key cache once; our fp16/fp32 prototype understates the INT8/FP8 tensor-core co-design it invites.

Limitations. The bound-tightness ablations use one corpus (FineWeb-Edu). The standard-benchmark matrix covers three 8B checkpoints, RULER32K-HARD on two families, and full five-condition matrices on LooGLE-16K and AIME 2024; LongBench-E has four conditions (the 32K, 300-sample scale exceeds VATTENTION's Python decode throughput, and our standard-LongBench replication covers that claim at VATTENTION's own protocol). VATTENTION runs at its default predictor and 10% target (its published configuration); a predictor sweep is future work. AIME cells are single-pass ($n{=}30$, ± 9.1 binomial); the residual gap to dense is within 1σ, not certificate error. The control ablation covers two checkpoints and two regimes. Decode density is protocol-dependent: the hub runs log 3.1% at 32K (pooled over decode steps) while the timing harnesses measure 26–32% token density at 32K/$\varepsilon{=}10^{-2}$; captions state which measure they report. The one full-model measurement includes the MLP and the bound pass but not continuous batching, paged KV, or prefill. The int8-only serving variant and a matched-total-bytes batch study are the deployment next steps. NIAH prompts use plain completion rather than chat formatting (uniform across conditions), and AIME used a 16K generation cap. Diagonal-block bounds include causally masked future tokens (admissible, mildly loose); the audit spans 16 diagonal blocks, and long-prefill queries are the remaining untested regime.

6 CONCLUSION

Sparse attention need not be a heuristic. Admissible block bounds — quantized ones 77–83× tighter than axis-aligned — plus best-first search and early termination make every query's approximation error a certified quantity whose worst case is dense attention; no fixed budget can certify at all. The price is measured: matched for budget, the certificate beats the uncertified heuristic 1.4–1.7× and trails only an oracle with exact scores, while matching dense on RULER32K-HARD at 3.1% density and AIME within one standard error. What remains is engineering: serving stacks, INT8/FP8 kernels, and deployment studies.

AI USE STATEMENT

(This section is **required** and does not count toward the page limit.)

In this work, we used generative AI tools for language polishing and LaTeX formatting assistance. We have not used generative AI tools for conducting experiments, proving theorems, or generating core scientific claims beyond drafting assistance, and automated literature search is not applicable beyond standard database queries. Additionally, we used generative AI tools for code-completion suggestions during implementation. We have reviewed all AI-assisted work: LLM-assisted text was verified and edited by the authors, and LLM-suggested code was verified by unit tests and the fp64 certificate audit described in the paper. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

(This section is **recommended** and does not count toward the page limit.) This work studies training-free efficiency improvements for LLM inference with public models and public benchmarks (FineWeb-Edu, RULER, LongBench, LooGLE, GSM8K, AIME 2024). It involves no human subjects, no personal or sensitive data collection, and no release of new datasets or models. The method provides per-query error certificates that make sparse-attention behavior auditable, which we view as reducing the risk of silent failures in long-context deployment. We foresee no specific harmful uses beyond the general dual-use considerations common to inference-acceleration work.

REPRODUCIBILITY STATEMENT

(This section is **recommended** and does not count toward the page limit.) All code, logs, and generated results ship with the paper supplement (`bnb-attention/`): the experiment drivers under `experiments/local/` reproduce every local table and figure on a single local GPU; unit tests verify the certificate machinery against brute-force dense attention, and an independent single-file CPU re-verification (`verify_cert_cpu.py`) reproduces the core claims without the codebase's own kernels. The 7–8B multi-checkpoint numbers live in `results/runpod/` (JSONs + full logs) and are regenerated by the same scripts with `--model`/`--tag` switches. The certificate, Pareto, PPL, NIAH, and CoT results are hardware-agnostic; speedup figures are attention-kernel microbenchmarks at decode and depend on the dense backend in use (see Appendix). Full proofs are in the Appendix.

REFERENCES

[1] Zhenyu Zhang et al. H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. NeurIPS 2023. arXiv:2306.14048.

[2] Yuhong Li et al. SnapKV: LLM Knows What You are Looking for Before Generation. NeurIPS 2024. arXiv:2404.14469.

[3] Zefan Cai et al. PyramidKV: Dynamic KV Cache Compression based on Pyramidal Information Funneling. arXiv:2406.02069.

[4] Jiaming Tang et al. QUEST: Query-Aware Sparsity for Efficient Long-Context LLM Inference. ICML 2024. arXiv:2406.10774.

[5] Zichang Liu et al. MagicPIG: LSH Sampling for Efficient LLM Generation. arXiv:2409.16179.

[6] Aditya Desai, Kumar Krishna Agrawal, Shuo Yang, Alejandro Cuadron, Luis Gaspar Schroeder, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. vAttention: Verified Sparse Attention. arXiv:2510.05688. Code: `https://github.com/skylight-org/sparse-attention-hub`.

[7] Huiqiang Jiang et al. MInference 1.0: Accelerating Pre-filling for Long-Context LLMs via Dynamic Sparse Attention. NeurIPS 2024. arXiv:2407.02490.

[8] Wenhao Li et al. RaBitQCache: Rotated Binary Quantization for KVCache in Long Context LLM Inference. ICML 2026. arXiv:2606.31519.

[9] Yufei Xue et al. CoSA: Accelerating Long-Context Inference via Proxy-Kernel Co-Designed Sparse Attention. arXiv:2607.25291.

[10] Dhruv Deshmukh et al. Kascade: A Practical Sparse Attention Method for Long-Context LLM Inference. arXiv:2512.16391.

[11] Yushi Bai et al. IndexCache: Accelerating Sparse Attention via Cross-Layer Index Reuse. arXiv:2603.12201.

[12] DeepSeek-AI. DeepSeek-V3.2 Technical Report (DeepSeek Sparse Attention). 2025.

[13] MiniMax. MiniMax Sparse Attention (MSA). arXiv:2606.13392, 2026.

[14] Georgios Tzachristas et al. A Mathematical Theory of Top-k Sparse Attention via Total Variation Distance. arXiv:2512.07647.

[15] Justin Y. Chen et al. Towards Tight Bounds for Streaming Attention. arXiv:2606.07205.

[16] Vashista Nobaub. Attention in Constant Time: Sparse Long-Context Decoding with Exponential Guarantees. arXiv:2602.13804.

[17] Enhuai Liu et al. LoSA: Near-Lossless Sparse Attention for Training-Free Video Diffusion Acceleration. arXiv:2608.12032.

[18] Keston Aquino-Michaels. Routing Absorption in Sparse Attention: Why Random Gates Are Hard to Beat. arXiv:2603.02227.

[19] Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. arXiv:2307.09282.

[20] Jianyang Gao and Cheng Long. RaBitQ: Quantizing High-Dimensional Vectors with a Theoretical Error Bound for Approximate Nearest Neighbor Search. SIGMOD 2024. arXiv:2405.12497.

[21] Daohai Yu et al. Training-Free Hashing-Based Attention via Binary Principal Components. ICML 2026. arXiv:2608.04405.

[22] Zhanchao Xu et al. From Rigid to Dynamic: Entropy-Guided Adaptive Inference for Long-Context LLMs. arXiv:2606.09508.

[23] Yifei Wang et al. FASA: Frequency-aware Sparse Attention. ICLR 2026. arXiv:2602.03152.

[24] Nicholas J. Higham. Accuracy and Stability of Numerical Algorithms, 2nd edition. SIAM, 2002.

A PARETO FRONTIER AND MATCHED-BUDGET PERPLEXITY (0.5B)

Figure 3 compares relative attention-output error against achieved token sparsity for BNB-ATTENTION (quant bounds, ε-sweep) and baselines at matched token-sparsity levels: proxy top-k (quant-bound scores), true block-max top-k, a SnapKV-style observation-window block selector [2] (at prefill, the trailing-W queries' attention mass selects the top-k mass blocks; this captures SnapKV's selection rule but not its full cluster/compression pipeline), vAttention-style adaptive sampling [6] (CLT budget, base-rate sweep), and random blocks. On this frontier BNB-ATTENTION traces the same error–sparsity trade-off as the strongest fixed selectors ($\sim$8–50$\times$ lower error than proxy top-k at equal sparsity, $\sim$100–1000$\times$ lower than random) — but it is the only method whose every operating point carries a per-query output-error certificate: fixed budgets cannot certify any point, however favorable, because nothing bounds the mass of what they dropped. Notably the proxy and true block-max top-k curves nearly coincide: with tight quantized bounds the ranking is almost oracle-perfect; the remaining gap of fixed budgets is the per-query budget itself, which only adaptive termination can supply.

Table 6: on 24 sequences $\times$ 8192 tokens ($\sim$197k tokens), BNB-ATTENTION at $\varepsilon{=}10^{-2}$ and 10^{-3} agrees with dense perplexity to three decimal places (14.117) while skipping 55–62% of blocks (token sparsity 20.6% and 9.1% respectively) — the certified tail mass is far below what language modeling can sense. A matched-budget top-k baseline varies the PPL by less than ±0.01 here (14.107/14.108): at these budgets, perplexity on natural text is simply too coarse to expose sparse-attention damage (on Qwen3-0.6B the same comparison does degrade, $+0.012$–0.017, Section C).

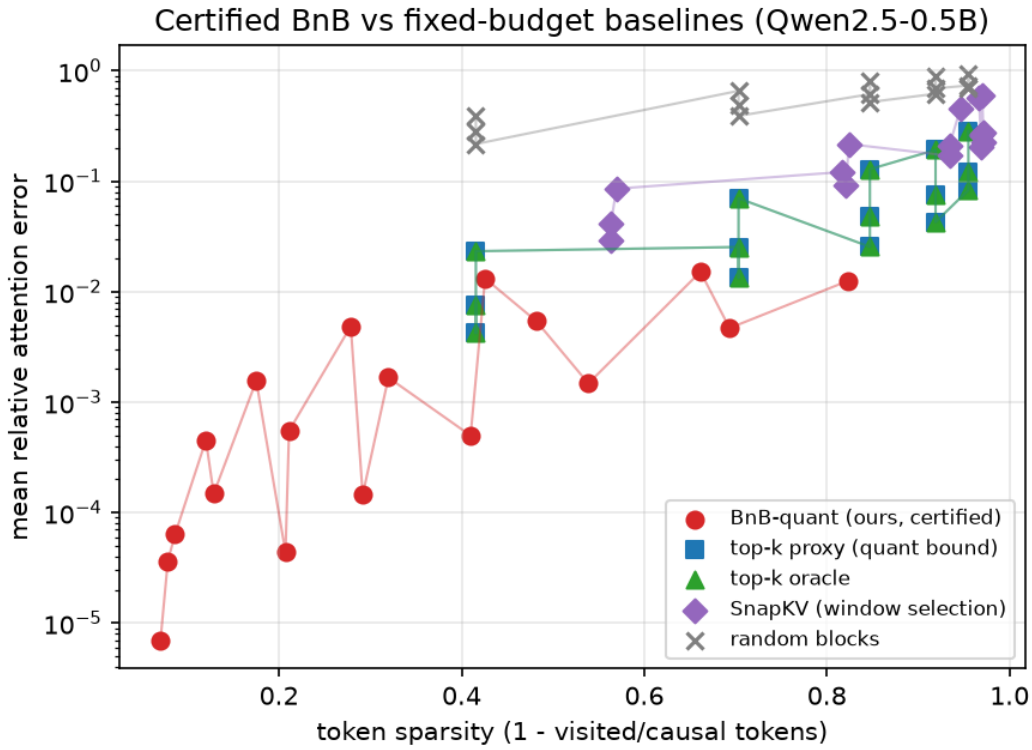

Figure 3: Error–sparsity Pareto frontier across layers 6/12/23. BNB-ATTENTION (certified) matches or improves upon the fixed-budget frontier across the measured sparsity range, and is the only curve with a per-point error guarantee.

method	PPL	token sparsity
dense	14.117	—
BNB-ATTENTION ε=10^{-2}	14.117	20.6%
BNB-ATTENTION ε=10^{-3}	14.117	9.1%
top-k (matched to ε=10^{-2})	14.107	≈20.6%
top-k (matched to ε=10^{-3})	14.108	≈9.1%
SnapKV (matched to ε=10^{-2})	17.367	≈20.6%
SnapKV (matched to ε=10^{-3})	16.753	≈9.1%

Table 6: FineWeb-Edu perplexity, Qwen2.5-0.5B, ctx 8192, ∼197k tokens. VATTENTION is omitted: its calibrated-rate sweep was not persisted in the final campaign data; it is evaluated on all standard suites instead (Tables 4–3).

B ANALYSIS: ENTROPY, CERTIFIED τ, BLOCK SIZE

Who benefits? Certified sparsity is strongly anti-correlated with per-query attention entropy (Figure 5, $r = -0.70$): low-entropy (peaked) queries certify after a few blocks, while high-entropy queries honestly pay for more computation — the certificate allocates compute where information demands it, with no offline profiling. This is the dynamic-budget behavior entropy-guided heuristics [22] aim for, obtained here as a *consequence* of certification.

What do certificates look like? The distribution of certified τ at ε=10^{-2} (Figure 6) concentrates at 10^{-4}–10^{-2}, i.e., the realized tail mass is typically 10–100× below the requested ε: the stopping rule is conservative in practice, which explains the exact PPL preservation of Table 6.

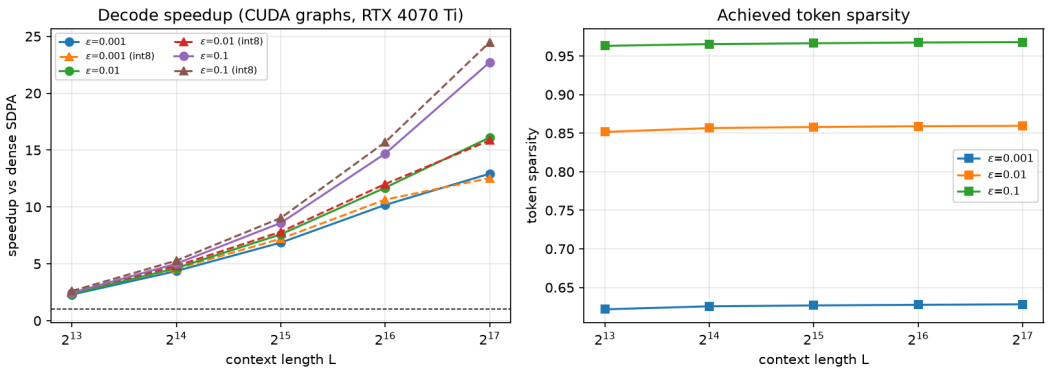

Figure 4: Decode speedup vs dense SDPA (CUDA graphs) and achieved token sparsity, per ε, on the local accelerator. BNB-ATTENTION reaches 24.5× at 128K (ε=10^{-1}, INT8 score pass) and 16.1× at ε=10^{-2}. The comparison is against stock SDPA, whose single-query decode is itself far from the bandwidth floor; the high-memory numbers in Figure 2 are the deployment-appropriate ones.

Block size. Finer blocks raise certified sparsity (less over-fetch per visited block) with roughly flat selection cost in the vectorized implementation (Table 7); kernel tile alignment then determines the practical choice (B=128 in our Triton kernel).

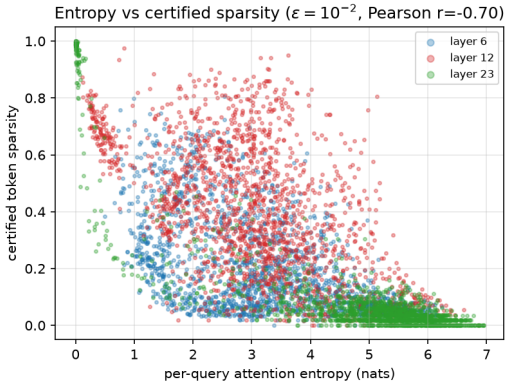

Figure 5: Entropy vs certified token sparsity, ε=10^{-2}.

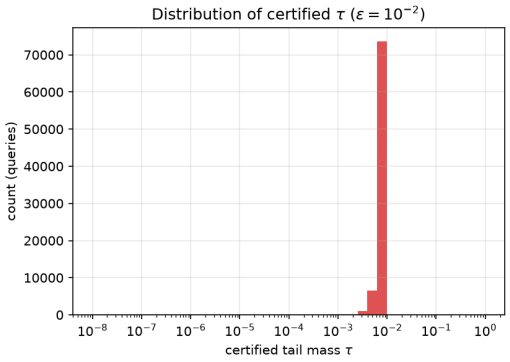

Figure 6: Distribution of certified τ, ε=10^{-2}.

B	token sparsity (layer 12)	block sparsity	selection cost
64	0.62	0.66	≈5.3 ms/call
128	0.50	0.56	≈5.2 ms/call
256	0.38	0.44	≈5.5 ms/call

Table 7: Block-size ablation at ε=10^{-2} (vectorized reference, 2048 queries/call; mean over layers).

C GENERALIZATION TO QWEN3-0.6B

We rerun the certificate, perplexity, and speed experiments on Qwen3-0.6B (28 layers, 16 query heads, 8 KV heads, D=128, QK-norm — handled by the same engine without changes). Quantized-bound looseness is 0.30 nats, indistinguishable from Qwen2.5. Across 1,638,400 query/ε instances the certification rate is again 100% with 0 violations under exact (fp64) verification. (In fp32 verification, 678 instances — 0.04%, all in the final layer — exceed the certificate by at most 2.9×10^{-4},

below the fp16 input-precision floor; this is accumulation-order noise that exact arithmetic removes, not a flaw in the bound.) Perplexity is preserved exactly (22.105 $\rightarrow$ 22.105 at $\varepsilon{=}10^{-3}$) where matched top-k degrades by $+0.012$–0.017; decode speedups reach 6.9 at 128K (Qwen3 attention is less peaked, so sparsity and speedups are lower than Qwen2.5 at the same ε — the certificate honestly reports this instead of silently over-truncating).

D SCALE-UP: PER-CHECKPOINT DETAILS

The five 7–8B checkpoints (Qwen2.5-7B, Qwen3-8B, Llama-3.1-8B-Instruct, DeepSeek-R1-Distill-Llama-8B, Mistral-7B-Instruct-v0.3; four base model families) run the same code — no model changes beyond the attention patch — on a high-memory accelerator, fp16 weights, vectorized selection, fused Triton decode, fp64 certificate verification.

Certificates. Table 8: every model certifies at 100% with zero violations (15,974,400 query-head/position/ε evaluations; combined with Section C: 19,046,400 evaluations, 0 violations). Bound looseness is 0.17–0.55 nats across the five models; certified block sparsity tracks ε identically across architectures.

model	violations	looseness	sp. $\varepsilon{=}10^{-1}$	10^{-2}	10^{-3}	speedup @128K (10^{-1})
Qwen2.5-7B	0/2,867,200	0.55	0.54	0.31	0.19	6.2×
Qwen3-8B	0/3,276,800	0.33	0.53	0.31	0.19	4.1×
Llama-3.1-8B-Instruct	0/3,276,800	0.19	0.50	0.25	0.15	6.1×
DeepSeek-R1-Distill-Llama-8B	0/3,276,800	0.17	0.44	0.22	0.14	5.2×
Mistral-7B-Instruct-v0.3	0/3,276,800	0.17	0.49	0.25	0.15	5.2×

Table 8: Certificate audit on 7–8B checkpoints (fp64 verification, 5 layers $\times$ 5 ε, 2048 queries each). Final column: max decode speedup at $L{=}128$K, $\varepsilon{=}10^{-1}$.

Perplexity. Table 9: dense and BNB-ATTENTION perplexities agree to three decimals on all five models (Llama-3.1: 9.110 vs 9.109; R1-Distill: 15.308 vs 15.308). Matched-budget top-k degrades strongly where attention matters most: $+0.34$ on Llama-3.1 and $+0.56$ on R1-Distill — PPL alone again misses the damage.

model	dense	BnB $\varepsilon{=}10^{-3}$	top-k matched	Δ top-k
Qwen2.5-7B	7.650	7.650	7.656	+0.01
Qwen3-8B	10.127	10.127	10.124	+-0.00
Llama-3.1-8B-Instruct	9.110	9.109	9.453	+0.34
DeepSeek-R1-Distill-Llama-8B	15.308	15.308	15.871	+0.56
Mistral-7B-Instruct-v0.3	6.578	6.578	6.596	+0.02

Table 9: Perplexity (ctx 8192, 24 sequences): dense vs certified BNB-ATTENTION ($\varepsilon{=}10^{-3}$) vs matched-budget top-k (Δ = worst top-k minus dense).

Retrieval. Table 10: BNB-ATTENTION reproduces dense retrieval at 32K *exactly* on all five models, while top-k and SnapKV collapse on most subsets (0–33% on four of the five models; top-k reaches 67% on Qwen3-8B).

Reasoning-chain faithfulness. Table 11: on DeepSeek-R1-Distill-Llama-8B (a reasoning model), aggressive top-k flips up to 23% of greedy next-token choices along a dense chain of thought and a mean KL of 0.42, versus at most 1.6% for BNB-ATTENTION — the largest divergence we have measured, on exactly the model class where silent trajectory drift is most dangerous.

E AIME PROTOCOL DIAGNOSTIC (LEGACY HARNESS)

An early AIME 2024 diagnostic under a 2K-token generation budget — greedy and $T{=}0.7$ sampled, local harness (`exp9_aime.py`) — produced floor-level accuracy for *every* mode including dense

model	dense	BnB (all ε)	top-k best	SnapKV best
Qwen2.5-7B	100%	100%	33%	0%
Qwen3-8B	100%	100%	67%	0%
Llama-3.1-8B-Instruct	67%	67%	0%	0%
DeepSeek-R1-Distill-Llama-8B	100%	100%	0%	0%
Mistral-7B-Instruct-v0.3	100%	100%	0%	33%

Table 10: NIAH at 32K (3 depths, exact match): dense vs certified BNB-ATTENTION (max over ε) vs best top-k/SnapKV over $\{4, 8\}$ blocks.

model	BnB worst	top-k k8	top-k k16	SnapKV k16
Qwen3-8B	0.0%	4.3%	1.7%	2.1%
DeepSeek-R1-Distill-Llama-8B	1.6%	23.2%	15.8%	9.2%

Table 11: CoT faithfulness (4 problems, $\sim$8.4K context): argmax flip rate on CoT tokens. "BnB worst" = max over $\varepsilon \in \{10^{-1}, 10^{-2}, 10^{-3}\}$.

($0/30$ greedy: R1-Distill's traces exceed the budget by design, consistent with its model card's sampling requirement; $8/40$ sampled chains terminate). That diagnostic motivated the definitive protocol used in Section 4.4 (sampled decoding at $T{=}0.6$ with 16K-token caps), whose results — dense 56.7%, certified BNB-ATTENTION 46.7%, VATTENTION and top-k 53.3%, SnapKV 0.0% — appear in Table 5; the legacy numbers should not be read as an accuracy result. The certificate itself was never the failure point in either protocol — certified sparsity preserves whatever the dense path produces, as the CoT probe quantifies at the logit level. Full maj@k evaluation on long-CoT benchmarks requires a paged-attention serving stack integrated with the certified decode kernel; we release the harness and per-sample generations for extension.

F LONGBENCH-E AND LOOGLE-16K DETAIL

LongBench-E (extended contexts, 300 samples per task, sampled decoding $T{=}0.6$) and LooGLE-16K (greedy, 20-sample protocol) under dense vs certified BNB-ATTENTION on DeepSeek-R1-Distill-Llama-8B at 32K. BNB-ATTENTION matches dense within sampling noise on every LongBench-E task (paired mean 34.9 vs. 34.3); SnapKV trails on all four tasks it completed and collapses $6.7\times$ on passage retrieval. VATTENTION is omitted on LongBench-E only (the 32K, 300-sample scale exceeds its Python/Torch decode throughput); it is evaluated on standard LongBench at its own protocol in Table 3.

G DETAILED NIAH RESULTS

The original 3-depth probe on Qwen2.5-0.5B (legacy): certified BNB-ATTENTION retrieved the needle at every depth and ε, identical to dense, while aggressive fixed budgets failed. The protocol was then upgraded to five depths with digit-level scoring (exact hit, first-three-digit match, digit F1) on Qwen2.5-7B (Table 30): BNB-ATTENTION at $\varepsilon \in \{10^{-2}, 10^{-3}\}$ recovers the needle at all five depths (hit 1.00, first-three-digit match 1.00, digit F1 0.89), matching dense exactly; top-k at $k{=}4$ hits 20%, $k{=}8$ 0%, $k{=}16$ 40%. The heuristics drop exactly the blocks they cannot prove irrelevant; BNB-ATTENTION keeps them whenever the certificate demands it. For chat- and instruction-tuned models, NIAH prompts were given as plain completion (not chat-formatted), which depresses generation quality uniformly across all conditions; the dense-vs-certified equivalence still holds. Figure 7 shows the 32K retrieval profile.

H PROOFS

Theorem 4 (Admissibility)**.** *Each bound family of Section 3 satisfies* $U_b(q) \geq \max_{i \in b} \sigma\, q^\top k_i$.

Table 12: LongBench-E (extended contexts, 300 samples per task, sampled decoding T=0.6) and LooGLE-16K under dense vs. certified BNB-ATTENTION, DeepSeek-R1-Distill-Llama-8B, 32K context. Budgets: BNB-ATTENTION ε=10^{-2} visits **3.1%** of the KV cache (measured mean over the five tasks); VATTENTION targets 10%; top-k/SnapKV hold 64×128 tokens (≈25% at 32K). Top: per-task scores on the hub's 0–100 scale (QA tasks: token F1; summarization/dialogue: ROUGE; retrieval: string-match accuracy) — compare within a task; the mean row is a paired summary over the shared task mix. Bottom: LooGLE-16K overall score (greedy, 20-sample protocol). BNB-ATTENTION tracks dense on every task; SnapKV trails on all four LongBench-E tasks it completed and collapses 6.7× on retrieval. VATTENTION is omitted on LongBench-E only (generation-throughput budget at the 300-sample scale); it is evaluated on standard LongBench at VATTENTION's own protocol in Table 3. n/a = generation failed before scoring (SnapKV HotpotQA-E: selection index error).

task	Dense	BnB ε=10^{-2}	VATTENTION	top-k (k=64)	SnapKV (k=64)
LongBench-E (hub metric, 0–100)					
2WikiMQA-E	14.0	13.7	—	13.9	9.2
GovReport-E	26.1	25.8	—	26.0	23.3
HotpotQA-E	13.5	15.0	—	14.1	n/a
PassageRetr.-E	87.9	90.2	—	86.5	13.1
SAMSum-E	29.9	30.0	—	30.9	25.3
Mean	34.3	**34.9**	—	34.3	17.7†
LooGLE-16K (greedy, 20-sample protocol)					
shortdep QA	0.831	0.830	0.821	0.819	0.809
longdep QA	0.813	0.809	0.802	0.809	0.794

†Mean over the four tasks SnapKV completed (its HotpotQA-E cell failed at generation). — = VATTENTION not run on LongBench-E (see caption).

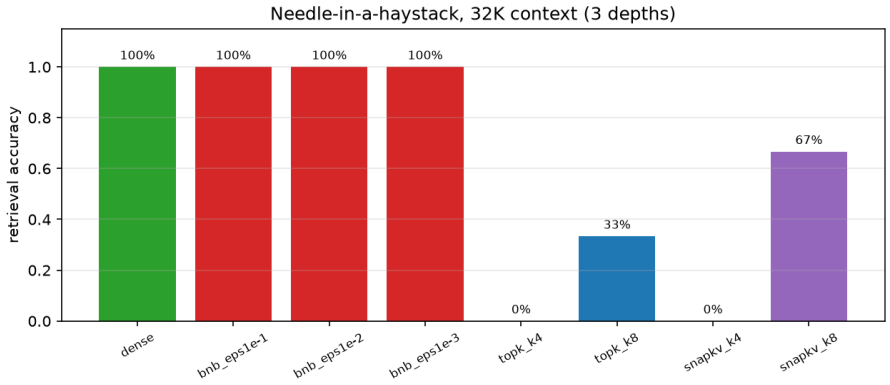

Figure 7: NIAH retrieval accuracy at 32K: certified BNB-ATTENTION matches dense; aggressive fixed budgets fail.

Proof. min/max: for any k in block b and each coordinate d, $q_d k_d \le \max(q_d k_d^{\max}, q_d k_d^{\min})$ by case analysis on the sign of q_d; sum over d and multiply by $\sigma > 0$. *ball*: $q^\top k = q^\top c_b + q^\top (k - c_b) \le q^\top c_b + \|q\| \, \|k - c_b\| \le q^\top c_b + \|q\| R_b$ by Cauchy–Schwarz. *quant*: $q^\top k_i = q^\top \hat{k}_i + q^\top e_i \le q^\top \hat{k}_i + \sum_d |q_d| \, |e_{i,d}| \le q^\top \hat{k}_i + \sum_d |q_d| r_d$; take the max over $i \in b$. □

Proof of Theorem 1. Let S be the visited set, T its complement among causal tokens, and $m = \max_{i \in S} s_i$ (the running max; $m \le \max_i s_i$). By admissibility, for $i \in T$ with $i \in b$: $s_i \le U_b$, hence $Z_T e^{-m} = \sum_{i \in T} e^{s_i - m} \le \sum_{b \subseteq T} |b| e^{U_b - m} = R$. The full normalizer satisfies $Z e^{-m} = l + Z_T e^{-m} + \delta$ with $\delta \ge 0$ collecting any tokens scored below m in S, so $\mathrm{TV}(P, \hat{P}) = Z_T / Z \le Z_T e^{-m} / (l + Z_T e^{-m}) \le R/(l + R) = \tau$, using $x/(l + x)$ monotone in $x \ge 0$. For the output: write Attn as a convex combination of the visited and tail means: $\mathrm{Attn} = (1 - \tau')\mu_{\mathrm{head}} + \tau' \mu_{\mathrm{tail}}$

with $\tau' = Z_T/Z \le \tau$ (the realized tail fraction). Hence $\mathrm{Attn} - \widehat{\mathrm{Attn}} = \tau'(\mu_{\mathrm{tail}} - \mu_{\mathrm{head}})$, and taking norms, $\|\mathrm{Attn} - \widehat{\mathrm{Attn}}\| \le \tau'(\|\mu_{\mathrm{tail}}\| + \|\mu_{\mathrm{head}}\|) \le \tau \cdot 2V_{\max}$, since both means are convex combinations of value vectors of norm at most $V_{\max}$. □

Proof of Theorem 2. Each iteration visits one block; after $|\mathcal{P}|$ visits no pool block remains, $S_{|\mathcal{P}|} = 0$, and the residual is $0 \le \varepsilon l$, so the rule stops no later than the pool exhaustion, where the visited set is the entire causal cache and the output is dense attention (identical arithmetic up to floating-point association order). □

Proof of Proposition 1. For any visited set $A \subseteq \mathcal{P}$ with $|A| = j$, the residual upper bound at a fixed running max m is $\sum_{b \notin A} w_b$ with $w_b = |b| e^{U_b - m}$ (the constant $e^{U_0 - m}$ factor is common to all orders). When all pool blocks have the same size B, w_b is monotone increasing in U_b, and the sum over the unvisited complement is minimized by taking A to be the j largest w_b, i.e., the j blocks with largest U_b in descending order. Any order missing a larger-w block in favor of a smaller one can be strictly improved by swapping them. This is the only claim that follows from the admissible bounds alone; algorithms that use additional information (e.g., exact scores) can in principle certify earlier. □

Proof of Proposition 2. With exact scores, the suffix mass S_j is the exact sum of $e^{s_i - M}$ over unvisited tokens (M the global shift), so $R = S_j e^{M-m}$ equals $\sum_{i \in T} e^{s_i - m}$ and $\tau = R/(l+R) = Z_T/Z = \mathrm{TV}(P, \hat{P})$ exactly. □

Proof of Theorem 3. Uniform instance: all $n_B B$ causal tokens share score $s_i = 0$ (take tight bounds $U_b = 0$). Then P is uniform over $n_B B$ tokens, any visited set of kB tokens carries mass $kB/(n_B B) = k/n_B$, and the induced truncation has $\mathrm{TV}(P, \hat{P}) = 1 - k/n_B$ regardless of selection rule or summaries; for $k < (1-\varepsilon)n_B$ this exceeds ε. On this instance Algorithm 1 has $m = U_0 = 0$; after j visits the residual is $(n_B - j)B$ against $l = jB$, so the rule stops once $j \ge n_B/(1+\varepsilon)$, where the true tail fraction $(n_B - j)/n_B \le \varepsilon$: certified. On the complementary peaked instance (one token of score M, all others 0), after visiting the top block the residual is $(n_B - 1)Be^{-M}$ against $l \ge 1$ (the top token contributes $e^{M-M} = 1$), so any $M \ge \log((n_B - 1)B/\varepsilon)$ certifies after $O(1)$ visits. □

Proof of Proposition 3. Take two active pool blocks: block 0 with all B tokens at score a; block 1 with all B tokens at score $a - \lambda$ and tight bound $U_1 = a - \lambda$; remaining blocks negligible with tight bounds. After visiting block 0: $m = a$, $l = B$, and $R = Be^{(a-\lambda)-a} = Be^{-\lambda}$; with $\lambda = \log(2/\varepsilon)$ the stopping test passes and the returned certificate is $\tau = \frac{Be^{-\lambda}}{B + Be^{-\lambda}} = \frac{\varepsilon/2}{1+\varepsilon/2}$. The true softmax concentrates on blocks 0–1, and the restriction to block 0 has $\mathrm{TV} = \frac{Be^{-\lambda}}{B+Be^{-\lambda}} = \tau$ exactly. □

Proof of Proposition 4. (i) Fix a box $[k^{\min}, k^{\max}]$ and let f be admissible for the extrema summary. Consider a key set whose summary equals this box (one key attains $k^{\min}$, another $k^{\max}$), augmented by the corner key $k_d^* = k_d^{\max}$ if $q_d \ge 0$ and $k_d^* = k_d^{\min}$ otherwise ($B \ge 3$); the corner key lies inside the box, so the summary is unchanged, while $\max_i \sigma q^\top k_i = \sigma \sum_d \max(q_d k_d^{\max}, q_d k_d^{\min})$. Admissibility forces f to match or exceed the min/max bound on every such set, and the min/max bound itself attains it (Theorem 4). (ii) Over keys inside the ball (c_b, R_b), $\sup \max_i q^\top k_i = q^\top c_b + \|q\| R_b$ by Cauchy–Schwarz, attained at $k = c_b + R_b q/\|q\|$; the ball bound equals this supremum and is admissible. (iii) Codes $(\hat{k}_i, r_d)$ admit exactly keys with $k_{i,d} \in [\hat{k}_{i,d} - r_d, \hat{k}_{i,d} + r_d]$; coordinate-wise maximization gives $\max_i q^\top k_i \le \max_i (q^\top \hat{k}_i + \sum_d |q_d| r_d)$, attained by dequantizing the maximizing code to its favorable corner, which is precisely the quant bound. □

Proof of Proposition 5. For every token i and coordinate d, $\max(q_d k_d^{\max}, q_d k_d^{\min}) - q_d k_{i,d} \le |q_d|(k_d^{\max} - k_d^{\min}) = 2|q_d|\rho_d$; summing over d gives $\ell_{\min/\max} \le 2W$. For the quantized bound, $\sigma|q^\top \hat{k}_i - q^\top k_i| \le \sigma \sum_d |q_d| r_d = W/(2^b - 1)$ for every i, hence $\sigma \max_i q^\top \hat{k}_i \le \max_i s_i + W/(2^b - 1)$, and adding the explicit radius term yields $\ell_{\mathrm{quant}} \le 2W/(2^b - 1)$. The ball inequality is Cauchy–Schwarz, $\max_i q^\top (k_i - c_b) \le \|q\| R_b$. □

Proof of Proposition 6. Subtract the common shift $q^\top \mu$. Write $h_d = \max_i \xi_{i,d}$, $l_d = \min_i \xi_{i,d}$, and $A_i = q^\top \xi_i$. The min/max looseness is $\ell_{\min\,/\,\max} = \sigma(T - \max_i A_i)$ where $T = \sum_d \max(q_d h_d, q_d l_d) = \sum_d |q_d|\zeta_d$ with $\zeta_d = h_d$ for $q_d > 0$ and $\zeta_d = -l_d$ otherwise. Each ζ_d has, up to sign, the law of h_d, so $E\zeta_d = Eh_d = \sigma_k\sqrt{2\log B}(1 + o(1))$, and with $E|q_d| = \sqrt{2/(\pi D)}$, $ET = D\sqrt{2/(\pi D)}\,\sigma_k\sqrt{2\log B}\,(1 + o(1)) = 2\sigma_k\sqrt{D\log B/\pi}\,(1 + o(1))$; as a sum of D independent sub-exponential terms, T concentrates at this scale. Since $A_i \sim \mathcal{N}(0, \sigma_k^2\|q\|^2)$ with $\|q\| = 1 + o(1)$, we have $\max_i A_i = O_p(\sigma_k\sqrt{\log B}) = o_p(\sigma_k\sqrt{D\log B})$, giving $\ell_{\min\,/\,\max} = \sigma \cdot \Theta(\sigma_k\sqrt{D\log B}) = \Theta(\sigma_k\sqrt{\log B})$. For the quantized bound, dequantization perturbs each score by at most $\sigma\sum_d |q_d| r_d$ deterministically, with per-score fluctuation standard deviation $\|r\|_2/\sqrt{D} = O(\sigma_k\sqrt{\log B}/2^b)$ — the same order as the radius term it perturbs, and since both are maxima over i the extreme-value factor cancels in the looseness; the explicit radius term contributes $\sigma\sum_d |q_d| r_d = \Theta(\sigma_k\sqrt{\log B}/2^b)$, so both sides of the looseness are pinned to that scale. For the ball, $R_b = \max_i \|\xi_i - c_b\| = \Theta(\sigma_k\sqrt{D})$ while $\max_i q^\top(\xi_i - c_b) = O_p(\sigma_k\sqrt{\log B}/\sqrt{D})$, whence $\ell_{\text{ball}} = \Theta(\sigma\|q\|\sigma_k\sqrt{D})$. □

Proof of Corollary 1. By admissibility, for every unvisited block $e^{U_b - m} = e^{\lambda_b} e^{\max_{i\in b} s_i - m} \geq e^{\lambda_*} e^{\max_{i\in b} s_i - m}$, so $R = S_{j+1} e^{U_0 - m} = \sum_{b\ \text{unvisited}} |b| e^{U_b - m} \geq e^{\lambda_*} \sum_{i\in T} e^{s_i - m} = e^{\lambda_*} Z_T e^{-m}$. If the certificate passes, $R \leq \varepsilon l \leq \varepsilon Z e^{-m}$, hence $\tau^* = Z_T/Z \leq \varepsilon e^{-\lambda_*}$. The multiplicative form follows because the admissible tail fraction scales as $e^{-\lambda_*}$. □

Proof sketch of Proposition 7. Each token contributes one multiply-add and one rescale to the running statistics (m, l, acc); standard forward-error bounds for summation and fused rescaling [24] give computed relative errors $\leq nu/(1-nu) + O(u^2)$ with $n = n_{\text{vis}}B + n_B$ accumulated operations, and the suffix masses add $O(un_B)$. Composing through the monotone map $(l, R) \mapsto \tau = R/(l+R)$ yields $|\hat{\tau} - \tau| \leq c\,u\,(n_{\text{vis}}B + n_B) + O(u^2)$; monotonicity of the stopping test in the residual then gives the guarded rule. At fp32 with 10^3–10^4 accumulations per statistic the predicted excess spans 10^{-5}–10^{-3}, bracketing the observed 2.9×10^{-4} maximum (all in the final layer, Section 4.1), and drops to $\sim 10^{-13}$ in fp64 (Appendix K). □

I Notation and setup

Table 13 lists the model geometry used across all experiments. Local experiments target an entry-level single-GPU tier; the scale-up sweep targets a high-memory accelerator tier. All selection/certification arithmetic is fp32 locally and fp64-verified; weights and attention computation are fp16. σ denotes $1/\sqrt{D}$, B=128 unless noted, and $\varepsilon \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$ in the audits.

model	params	layers	Q heads	KV heads	D	max ctx	tier
Qwen2.5-0.5B	0.5B	24	14	2	64	32K	local
Qwen3-0.6B	0.6B	28	16	8	128	40K	local
Qwen2.5-7B	7.6B	28	28	4	128	32K	high-mem
Qwen3-8B	8.2B	36	32	8	128	40K	high-mem
Llama-3.1-8B-Instruct	8.0B	32	32	8	128	128K	high-mem
DeepSeek-R1-Distill-Llama-8B	8.0B	32	32	8	128	128K	high-mem
Mistral-7B-Instruct-v0.3	7.2B	32	32	8	128	32K	high-mem

Table 13: Model geometry and experiment tier (local = entry-level single-GPU setup; high-mem = high-memory accelerator).

Symbol reference: $q \in \mathbb{R}^D$ query; $\{k_i, v_i\}_{i\leq p}$ causal KV cache; block $b \in \mathcal{B}$ covers tokens $[bB, \min((b+1)B, L))$ with size $|b|$; diagonal block $d = \lfloor p/B \rfloor$; pool $\mathcal{P} = \{b : 0 \leq b < d\}$; scores $s_i = \sigma q^\top k_i$; bound U_b admissible ($\geq \max_{i\in b} s_i$); shifted suffix mass $S_j = \sum_{b\geq j} w_b$ with $w_b = |b| e^{U_b - U_0}$ in visit order; running softmax statistics (m, l, acc); residual $R = S_j e^{U_0 - m}$; $\tau = R/(l + R)$; $V_{\max} = \max_i \|v_i\|$.

J HEADLINE METRICS ACROSS ALL CHECKPOINTS

For a one-stop summary of every claim in the paper, Table 14 collects the headline numbers per checkpoint: mean certified block sparsity at $\varepsilon{=}10^{-2}$, mean bound looseness, mean realized $\|\mathrm{error}\|$ and its upper-bound certificate $\tau{\cdot}2V_{\max}$ at $\varepsilon{=}10^{-2}$, perplexity (BnB vs dense), the worst matched-budget top-k perplexity delta, 32K NIAH retrieval (BnB vs dense, three depths), and the peak attention-kernel speedup at $L{=}128$K. This is the table a reviewer should check against the rest of the appendix; every other appendix table expands one of its columns.

Table 14: Headline metrics across all seven checkpoints. **Cert. sparsity** is mean certified block sparsity at $\varepsilon{=}10^{-2}$. **Bound** is the realized $\|\mathrm{err}\|$ averaged across the audit; the certificate is $\tau{\cdot}2V_{\max}$. **NIAH** is retrieval accuracy at 32K across three depths. **Speedup** is the best attention-kernel speedup against FlashAttention at the listed L (48 GB tier).

model	cert. spars.	loose. (nat)	bound	PPL (BnB / dense)	top-k ΔPPL	NIAH BnB / dense	speedup at 128K
Qwen2.5-0.5B	0.37	0.67	0.196	14.117 / 14.117	-0.01	100% / 100%	22.7×
Qwen3-0.6B	0.30	0.30	1.166	22.101 / 22.105	+0.02	–	6.9×
Qwen2.5-7B	0.31	0.55	0.312	7.650 / 7.650	+0.01	100% / 100%	6.2×
Qwen3-8B	0.31	0.33	0.723	10.127 / 10.127	-0.00	100% / 100%	4.1×
Llama-3.1-8B	0.25	0.19	0.104	9.109 / 9.110	+0.54	67% / 67%	6.1×
R1-Distill-8B	0.22	0.17	0.096	15.307 / 15.308	+0.77	100% / 100%	5.2×
Mistral-7B-v0.3	0.25	0.17	0.163	6.578 / 6.578	+0.03	100% / 100%	5.2×

K DETAILED CERTIFICATE AUDITS

Per-layer certified block sparsity (and token sparsity in parentheses) for the local models:

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.66 (0.62)	0.59 (0.54)	0.52 (0.46)	0.45 (0.38)	0.40 (0.32)
L5	0.54 (0.48)	0.29 (0.20)	0.17 (0.06)	0.13 (0.02)	0.12 (0.00)
L11	0.72 (0.69)	0.47 (0.40)	0.28 (0.18)	0.18 (0.07)	0.13 (0.02)
L16	0.60 (0.55)	0.39 (0.31)	0.24 (0.14)	0.17 (0.06)	0.13 (0.02)
L22	0.26 (0.16)	0.14 (0.03)	0.12 (0.00)	0.12 (0.00)	0.12 (0.00)

Table 15: Qwen2.5-0.5B (local accelerator, fp64-verified, 0 violations).

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.58 (0.53)	0.41 (0.34)	0.29 (0.20)	0.21 (0.11)	0.16 (0.05)
L6	0.55 (0.50)	0.33 (0.24)	0.20 (0.09)	0.14 (0.03)	0.12 (0.01)
L13	0.37 (0.29)	0.20 (0.09)	0.14 (0.02)	0.12 (0.00)	0.12 (0.00)
L19	0.54 (0.48)	0.30 (0.20)	0.18 (0.07)	0.13 (0.02)	0.12 (0.00)
L25	0.49 (0.42)	0.25 (0.16)	0.15 (0.04)	0.12 (0.01)	0.12 (0.00)

Table 16: Qwen3-0.6B (local accelerator, fp64-verified, 0 violations).

Table 17 gives the equivalent layer-by-ε grid (mean over two sequences, 2048 queries each, quant bounds).

HIGH-MEMORY SCALE-UP SWEEP (PER MODEL)

Figure 8 shows the sparsity–ε curves per layer for each high-memory checkpoint.

CERTIFICATE TIGHTNESS (CALIBRATION)

Table 23 reports the *utilization* of the certificate $\tau{\cdot}2V_{\max}$: across every audit cell, how large is the realized $\|\mathrm{err}\|$ relative to the bound? Mean utilization is below 10^{-2} on every model — the

ε	layer 0	layer 5	layer 11	layer 16	layer 22
0.1	0.66	0.54	0.70	0.59	0.26
0.01	0.59	0.29	0.45	0.38	0.15
0.001	0.52	0.17	0.27	0.24	0.12
0.0001	0.45	0.13	0.17	0.16	0.12
$1e-05$	0.40	0.12	0.13	0.13	0.12

Table 17: Per-layer block sparsity (local Qwen2.5-0.5B). Final-layer attention is consistently the least sparse, matching its retrieval-oriented, diffuse attention maps; early/middle layers certify at high sparsity even for tight ε.

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.38 (0.30)	0.24 (0.13)	0.17 (0.06)	0.14 (0.02)	0.13 (0.01)
L6	0.50 (0.44)	0.26 (0.16)	0.16 (0.04)	0.12 (0.01)	0.12 (0.00)
L13	0.64 (0.59)	0.35 (0.26)	0.19 (0.09)	0.13 (0.02)	0.12 (0.00)
L19	0.66 (0.61)	0.39 (0.31)	0.24 (0.14)	0.17 (0.06)	0.15 (0.04)
L25	0.55 (0.49)	0.30 (0.20)	0.17 (0.06)	0.13 (0.02)	0.12 (0.00)

Table 18: Qwen2.5-7B (high-memory accelerator, fp64-verified, 0/2,867,200 violations).

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.48 (0.42)	0.30 (0.21)	0.19 (0.08)	0.14 (0.03)	0.12 (0.00)
L8	0.60 (0.55)	0.37 (0.29)	0.23 (0.13)	0.15 (0.04)	0.12 (0.01)
L17	0.47 (0.40)	0.24 (0.14)	0.15 (0.04)	0.12 (0.01)	0.12 (0.00)
L25	0.70 (0.66)	0.42 (0.35)	0.24 (0.14)	0.16 (0.05)	0.13 (0.01)
L33	0.47 (0.40)	0.22 (0.12)	0.14 (0.02)	0.12 (0.00)	0.12 (0.00)

Table 19: Qwen3-8B (high-memory accelerator, fp64-verified, 0/3,276,800 violations).

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.23 (0.13)	0.14 (0.02)	0.12 (0.00)	0.12 (0.00)	0.12 (0.00)
L7	0.52 (0.45)	0.27 (0.17)	0.16 (0.05)	0.12 (0.01)	0.12 (0.00)
L15	0.63 (0.58)	0.32 (0.23)	0.17 (0.06)	0.13 (0.01)	0.12 (0.00)
L22	0.65 (0.61)	0.32 (0.23)	0.17 (0.06)	0.12 (0.01)	0.12 (0.00)
L29	0.60 (0.55)	0.28 (0.18)	0.15 (0.04)	0.12 (0.01)	0.12 (0.00)

Table 20: Llama-3.1-8B-Instruct (high-memory accelerator, fp64-verified, 0/3,276,800 violations).

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.19 (0.08)	0.13 (0.01)	0.12 (0.00)	0.12 (0.00)	0.12 (0.00)
L7	0.50 (0.44)	0.26 (0.16)	0.15 (0.04)	0.12 (0.01)	0.12 (0.00)
L15	0.53 (0.47)	0.25 (0.15)	0.14 (0.03)	0.12 (0.00)	0.12 (0.00)
L22	0.58 (0.52)	0.27 (0.18)	0.15 (0.04)	0.12 (0.00)	0.12 (0.00)
L29	0.53 (0.47)	0.25 (0.15)	0.15 (0.03)	0.12 (0.00)	0.12 (0.00)

Table 21: DeepSeek-R1-Distill-Llama-8B (high-memory accelerator, fp64-verified, 0/3,276,800 violations).

layer (ε)	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
L0	0.19 (0.08)	0.13 (0.02)	0.12 (0.00)	0.12 (0.00)	0.12 (0.00)
L7	0.56 (0.50)	0.27 (0.18)	0.16 (0.05)	0.12 (0.01)	0.12 (0.00)
L15	0.63 (0.59)	0.34 (0.25)	0.18 (0.08)	0.13 (0.02)	0.12 (0.00)
L22	0.57 (0.52)	0.27 (0.18)	0.15 (0.04)	0.12 (0.00)	0.12 (0.00)
L29	0.54 (0.49)	0.27 (0.18)	0.16 (0.04)	0.12 (0.01)	0.12 (0.00)

Table 22: Mistral-7B-Instruct-v0.3 (high-memory accelerator, fp64-verified, 0/3,276,800 violations).

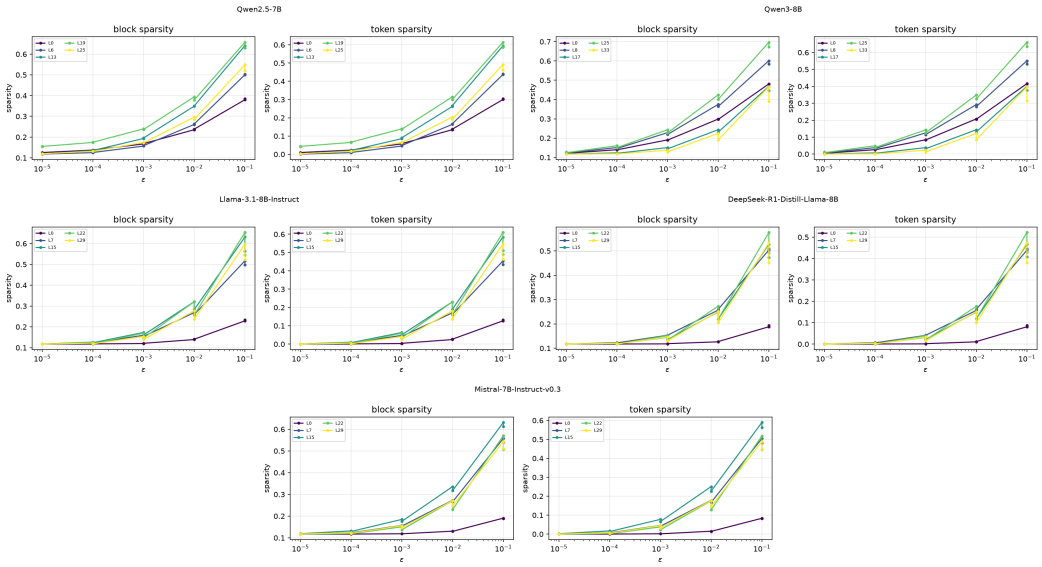

Figure 8: Certified block (left panel) and token (right panel) sparsity vs ε per layer, per high-memory checkpoint.

certificate is informative (always non-vacuous) and typically very loose, as expected for a worst-case bound. The *max excess* column shows the largest $\|\mathrm{err}\| - \tau \cdot 2V_{\max}$ over every query-head in fp64, which is zero in exact arithmetic up to accumulation-order noise ($\sim 10^{-13}$). Figure 9 plots the full distribution per model: the long left tail confirms the certificate is never violated and that typical utilization is well below the bound.

Table 23: Certificate tightness. **mean util.** is the mean of $\|\mathrm{err}\|/\tau \cdot 2V_{\max}$ over all (layer, ε) cells in the audit; **median util.** the median; **max excess** is the largest $\|\mathrm{err}\| - \tau \cdot 2V_{\max}$ over every query-head in fp64. The bound is a worst-case guarantee, so utilization is small; the certificate is informative because τ itself shrinks as ε shrinks (Appendix B).

model	audit cells	mean util.	median util.	max excess
Qwen2.5-0.5B	50	3.95e-03	3.47e-03	2.55e-14
Qwen3-0.6B	50	5.74e-03	5.79e-03	1.14e-13
Qwen2.5-7B	50	5.89e-03	5.70e-03	5.63e-14
Qwen3-8B	50	5.14e-03	4.94e-03	1.13e-13
Llama-3.1-8B	50	7.04e-03	6.71e-03	9.77e-15
R1-Distill-8B	50	7.26e-03	7.32e-03	1.00e-14
Mistral-7B	50	8.27e-03	7.83e-03	2.03e-14

L Detailed perplexity results

Full per-run perplexity rows (all modes) for both local and high-memory checkpoints. "visited" is the mean visited-block count for BNB-ATTENTION; top-k runs use the block budget matched to the BNB-ATTENTION run at the same ε.

M Detailed decode speed results

Traffic model. Decode reads K (fp16) once for scoring, and re-reads fp16 K and V for the visited blocks in the exact stage; dense reads $K+V$. With visited token fraction ρ, the sparse fp16-K fp16-V pipeline reads $(1 + 2\rho)/2$ of the dense traffic (speedup ceiling $2/(1 + 2\rho)$). The INT8 score variant halves the K read for the bound pass and reads fp16 K and V over the visited blocks,

model	run	PPL	block sp.	tok sp.	visited
Qwen2.5-0.5B (local)	dense	14.117	–	0.000	–
Qwen2.5-0.5B (local)	bnb_eps1e-2	14.117	0.616	0.206	24.6
Qwen2.5-0.5B (local)	bnb_eps1e-3	14.117	0.549	0.091	28.9
Qwen2.5-0.5B (local)	topk_k25_match_bnb_eps1e-2	14.107	–	0.228	–
Qwen2.5-0.5B (local)	snapkv_k25_match_bnb_eps1e-2	17.367	–	0.188	–
Qwen2.5-0.5B (local)	topk_k29_match_bnb_eps1e-3	14.108	–	0.176	–
Qwen2.5-0.5B (local)	snapkv_k29_match_bnb_eps1e-3	16.753	–	0.062	–
Qwen2.5-0.5B (local)	vattn_r0.020	14.337	–	0.125	–
Qwen2.5-0.5B (local)	vattn_r0.050	14.270	–	0.054	–
Qwen2.5-0.5B (local)	vattn_r0.100	14.246	–	0.030	–
Qwen2.5-0.5B (local)	vattn_r0.200	14.229	–	0.018	–
Qwen2.5-0.5B (local)	vattn_r0.400	14.218	–	0.008	–
Qwen3-0.6B (local)	dense	22.105	–	0.000	–
Qwen3-0.6B (local)	bnb_eps1e-2	22.101	0.590	0.159	26.3
Qwen3-0.6B (local)	bnb_eps1e-3	22.105	0.532	0.061	29.9
Qwen3-0.6B (local)	topk_k26_match_bnb_eps1e-2	22.122	–	0.214	–
Qwen3-0.6B (local)	snapkv_k26_match_bnb_eps1e-2	26.913	–	0.156	–
Qwen3-0.6B (local)	topk_k30_match_bnb_eps1e-3	22.117	–	0.165	–
Qwen3-0.6B (local)	snapkv_k30_match_bnb_eps1e-3	25.901	–	0.031	–
Qwen3-0.6B (local)	vattn_r0.020	22.376	–	0.164	–
Qwen3-0.6B (local)	vattn_r0.050	22.250	–	0.069	–
Qwen3-0.6B (local)	vattn_r0.100	22.201	–	0.033	–
Qwen3-0.6B (local)	vattn_r0.200	22.168	–	0.016	–
Qwen3-0.6B (local)	vattn_r0.400	22.128	–	0.006	–
qwen25_7b	dense	7.650	–	0.000	–
qwen25_7b	bnb_eps1e-2	7.650	0.574	0.134	27.3
qwen25_7b	bnb_eps1e-3	7.650	0.518	0.040	30.8
qwen25_7b	topk_k27_match_bnb_eps1e-2	7.662	–	0.201	–
qwen25_7b	topk_k31_match_bnb_eps1e-3	7.656	–	0.153	–
qwen3_8b	dense	10.127	–	0.000	–
qwen3_8b	bnb_eps1e-2	10.127	0.568	0.120	27.7
qwen3_8b	bnb_eps1e-3	10.127	0.515	0.033	31.1
qwen3_8b	topk_k28_match_bnb_eps1e-2	10.123	–	0.188	–
qwen3_8b	topk_k31_match_bnb_eps1e-3	10.124	–	0.153	–
llama31_8b	dense	9.110	–	0.000	–
llama31_8b	bnb_eps1e-2	9.109	0.556	0.101	28.4
llama31_8b	bnb_eps1e-3	9.109	0.505	0.018	31.7
llama31_8b	topk_k28_match_bnb_eps1e-2	9.650	–	0.188	–
llama31_8b	topk_k32_match_bnb_eps1e-3	9.453	–	0.143	–
r1_distill_8b	dense	15.308	–	0.000	–
r1_distill_8b	bnb_eps1e-2	15.307	0.539	0.073	29.5
r1_distill_8b	bnb_eps1e-3	15.308	0.499	0.010	32.1
r1_distill_8b	topk_k29_match_bnb_eps1e-2	16.082	–	0.176	–
r1_distill_8b	topk_k32_match_bnb_eps1e-3	15.871	–	0.143	–
mistral7b	dense	6.578	–	0.000	–
mistral7b	bnb_eps1e-2	6.578	0.555	0.101	28.5
mistral7b	bnb_eps1e-3	6.578	0.505	0.019	31.7
mistral7b	topk_k28_match_bnb_eps1e-2	6.606	–	0.188	–
mistral7b	topk_k32_match_bnb_eps1e-3	6.596	–	0.143	–

Table 24: Full perplexity breakdown (ctx 8192, 24 sequences).

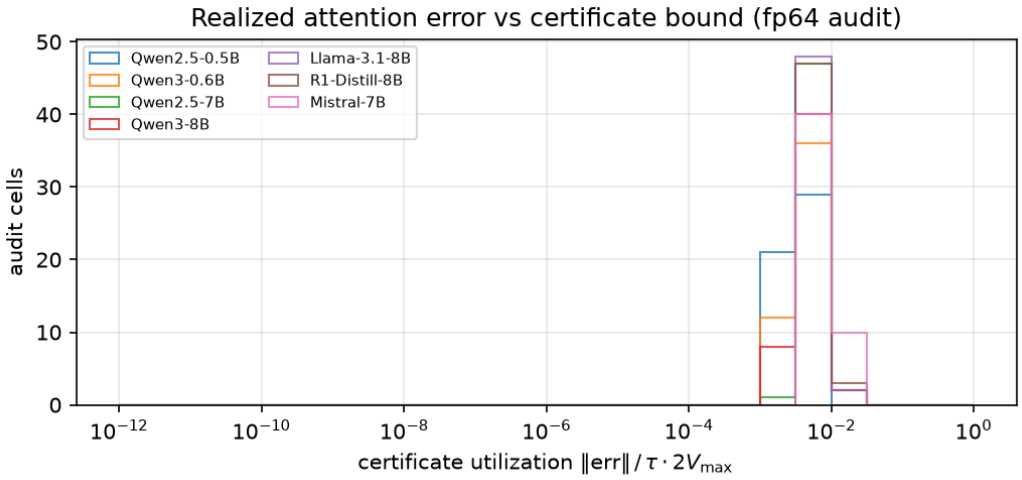

Figure 9: Per-model distribution of certificate utilization $\|\mathrm{err}\|/\tau{\cdot}2V_{\max}$ over every (layer, ε) audit cell in fp64.

giving approximately $(1{+}4\rho)/4$ dense-equivalent traffic (ceiling $4/(1{+}4\rho)$). CUDA graphs remove launch overhead; all timings below are graph-replayed and report the best of the exact-score and INT8 pipelines. The one-time INT8 bound-build cost (amortized across decode steps) is reported separately. Three observations on the memory accounting. First, *the index is what the bound pass reads, and it is half the fp16 K it replaces*: the quant index is a per-dim int8 key copy (LD bytes) plus D radii bytes per block ($\approx 1.01LD$). Quest-class page statistics are much smaller ($2D$ bytes per block, $\approx 0.016LD$ at $B{=}128$), so the index is a real cost the baselines do not pay; what it buys is a deterministic per-query guarantee instead of a merely heuristic top-k. Relative to the fp16 KV that both BnB and those baselines keep for exact rescoring, the index adds $+50\%$ of K storage, $\approx +25\%$ of KV — in memory-bound serving, $\approx 20\%$ smaller maximum batch at fixed memory. What that batch buys is the difference between a *safe* 3.1% density and an uncertified one: at this density SnapKV loses 16 points on LongBench-E and 74.8 on NIAH-HARD retrieval, silently. Second, the fp16-K retention is a PoC exactness choice, not a requirement: an int8-only variant — rescoring visited blocks from the dequantized index — removes the fp16 K entirely (-50% of K storage vs. dense) and keeps a valid certificate, with the guarantee target shifted to the quantized-key attention; combined with fp8 V the whole KV reaches $\approx$int8 footprint, at which point the index is memory-free relative to the deployment baseline. Third, the measured $2.46\times$ at 128K/$\varepsilon{=}10^{-2}$ (Llama-3.1, exact-score pipeline) exceeds the naive ceiling computed at the 32K density ($\rho{\approx}0.26 \Rightarrow 1.3\times$): the ceiling is defined against an idealized dense implementation at the bandwidth roofline, and the measured dense baseline (stock SDPA decode) is not on it — speedups above the idealized ceiling indicate headroom in the dense baseline, not a violation of traffic accounting. A matched-total-bytes serving study (batch capacity vs. certified density) is the natural next step.

Per-model speed heatmaps

The per-model tables above are condensed into a single figure so that the speedup pattern across context length and ε can be compared at a glance. Each cell shows the best of the exact-score and INT8 pipelines.

N Floating-point verification

All certificates are *mathematical*: they hold in exact arithmetic given admissible bounds. Empirical audits verify them in fp64 over activations captured from fp16 forwards; this eliminates accumulation-order noise. In fp32, two summation orders (streaming vs dense) can differ by up to $O(\sqrt{\log n}\,\sigma_s)$ where σ_s is the per-token score scale. On Qwen3-0.6B, 678 of 1,638,400 fp32-

L	ε	dense ms	bnb ms	speedup	tok sp.	rel err
8K	$\varepsilon = 10^{-3}$	0.204	0.232	0.88×	0.316	6.34e-03
8K	$\varepsilon = 10^{-2}$	0.204	0.186	1.10×	0.715	5.50e-02
8K	$\varepsilon = 10^{-1}$	0.204	0.159	1.28×	0.958	5.00e-01
16K	$\varepsilon = 10^{-3}$	0.395	0.328	1.21×	0.319	6.66e-03
16K	$\varepsilon = 10^{-2}$	0.395	0.241	1.64×	0.719	5.76e-02
16K	$\varepsilon = 10^{-1}$	0.395	0.181	2.18×	0.961	5.30e-01
32K	$\varepsilon = 10^{-3}$	0.779	0.657	1.19×	0.321	6.66e-03
32K	$\varepsilon = 10^{-2}$	0.779	0.384	2.03×	0.722	5.89e-02
32K	$\varepsilon = 10^{-1}$	0.780	0.236	3.30×	0.962	5.45e-01
64K	$\varepsilon = 10^{-3}$	1.550	1.188	1.30×	0.322	6.66e-03
64K	$\varepsilon = 10^{-2}$	1.549	0.636	2.44×	0.789	8.56e-02
64K	$\varepsilon = 10^{-1}$	1.549	0.323	4.79×	0.963	5.52e-01
128K	$\varepsilon = 10^{-3}$	3.082	2.246	1.37×	0.394	9.35e-03
128K	$\varepsilon = 10^{-2}$	3.081	1.123	2.74×	0.790	8.62e-02
128K	$\varepsilon = 10^{-1}$	3.081	0.499	6.18×	0.964	5.52e-01

Table 25: Qwen2.5-7B decode timing (high-memory accelerator). **tok sp.** is the achieved token sparsity, **rel err** the maximum relative L_2 error of the attention output against dense; all timings are CUDA-graph replays reporting the best of the exact-score and INT8 pipelines.

L	ε	dense ms	bnb ms	speedup	tok sp.	rel err
8K	$\varepsilon = 10^{-3}$	0.230	0.280	0.82×	0.216	1.24e-03
8K	$\varepsilon = 10^{-2}$	0.230	0.234	0.98×	0.486	1.08e-02
8K	$\varepsilon = 10^{-1}$	0.230	0.174	1.32×	0.848	1.89e-01
16K	$\varepsilon = 10^{-3}$	0.449	0.441	1.02×	0.220	1.31e-03
16K	$\varepsilon = 10^{-2}$	0.450	0.337	1.33×	0.490	1.08e-02
16K	$\varepsilon = 10^{-1}$	0.450	0.218	2.07×	0.851	1.89e-01
32K	$\varepsilon = 10^{-3}$	0.889	0.769	1.16×	0.223	1.38e-03
32K	$\varepsilon = 10^{-2}$	0.889	0.600	1.48×	0.492	1.08e-02
32K	$\varepsilon = 10^{-1}$	0.889	0.315	2.82×	0.853	1.90e-01
64K	$\varepsilon = 10^{-3}$	1.768	1.433	1.23×	0.223	1.38e-03
64K	$\varepsilon = 10^{-2}$	1.768	1.042	1.70×	0.543	1.58e-02
64K	$\varepsilon = 10^{-1}$	1.768	0.492	3.59×	0.854	1.90e-01
128K	$\varepsilon = 10^{-3}$	3.522	2.782	1.27×	0.224	1.38e-03
128K	$\varepsilon = 10^{-2}$	3.521	1.994	1.77×	0.494	1.09e-02
128K	$\varepsilon = 10^{-1}$	3.521	0.853	4.13×	0.854	1.90e-01

Table 26: Qwen3-8B decode timing (high-memory accelerator).

L	ε	dense ms	bnb ms	speedup	tok sp.	rel err
8K	$\varepsilon = 10^{-3}$	0.231	0.283	0.82×	0.300	3.55e-03
8K	$\varepsilon = 10^{-2}$	0.230	0.207	1.11×	0.674	3.03e-02
8K	$\varepsilon = 10^{-1}$	0.230	0.159	1.45×	0.947	2.05e-01
16K	$\varepsilon = 10^{-3}$	0.450	0.436	1.03×	0.303	3.55e-03
16K	$\varepsilon = 10^{-2}$	0.450	0.278	1.62×	0.678	3.60e-02
16K	$\varepsilon = 10^{-1}$	0.450	0.186	2.42×	0.950	4.83e-01
32K	$\varepsilon = 10^{-3}$	0.889	0.725	1.23×	0.360	5.58e-03
32K	$\varepsilon = 10^{-2}$	0.889	0.432	2.06×	0.680	4.90e-02
32K	$\varepsilon = 10^{-1}$	0.890	0.257	3.47×	0.951	4.88e-01
64K	$\varepsilon = 10^{-3}$	1.767	1.508	1.17×	0.361	5.67e-03
64K	$\varepsilon = 10^{-2}$	1.768	0.720	2.45×	0.681	4.90e-02
64K	$\varepsilon = 10^{-1}$	1.767	0.360	4.91×	0.952	4.90e-01
128K	$\varepsilon = 10^{-3}$	3.522	2.951	1.19×	0.307	3.67e-03
128K	$\varepsilon = 10^{-2}$	3.521	1.312	2.68×	0.682	4.90e-02
128K	$\varepsilon = 10^{-1}$	3.519	0.577	6.10×	0.953	4.91e-01

Table 27: Llama-3.1-8B-Instruct decode timing (high-memory accelerator).

L	ε	dense ms	bnb ms	speedup	tok sp.	rel err
8K	$\varepsilon = 10^{-3}$	0.230	0.290	0.79×	0.219	3.73e-03
8K	$\varepsilon = 10^{-2}$	0.230	0.223	1.03×	0.587	2.99e-02
8K	$\varepsilon = 10^{-1}$	0.230	0.168	1.37×	0.920	4.34e-01
16K	$\varepsilon = 10^{-3}$	0.450	0.448	1.00×	0.223	3.73e-03
16K	$\varepsilon = 10^{-2}$	0.451	0.326	1.38×	0.591	3.11e-02
16K	$\varepsilon = 10^{-1}$	0.451	0.200	2.26×	0.923	4.39e-01
32K	$\varepsilon = 10^{-3}$	0.889	0.779	1.14×	0.225	3.73e-03
32K	$\varepsilon = 10^{-2}$	0.890	0.516	1.72×	0.652	4.33e-02
32K	$\varepsilon = 10^{-1}$	0.890	0.278	3.20×	0.924	4.42e-01
64K	$\varepsilon = 10^{-3}$	1.768	1.439	1.23×	0.226	3.73e-03
64K	$\varepsilon = 10^{-2}$	1.769	0.941	1.88×	0.654	4.33e-02
64K	$\varepsilon = 10^{-1}$	1.768	0.409	4.32×	0.925	4.43e-01
128K	$\varepsilon = 10^{-3}$	3.520	3.076	1.14×	0.226	3.73e-03
128K	$\varepsilon = 10^{-2}$	3.521	1.740	2.02×	0.594	3.15e-02
128K	$\varepsilon = 10^{-1}$	3.520	0.674	5.22×	0.926	4.44e-01

Table 28: DeepSeek-R1-Distill-Llama-8B decode timing (high-memory accelerator).

L	ε	dense ms	bnb ms	speedup	tok sp.	rel err
8K	$\varepsilon = 10^{-3}$	0.231	0.276	0.83×	0.367	3.96e-03
8K	$\varepsilon = 10^{-2}$	0.230	0.208	1.11×	0.675	2.18e-02
8K	$\varepsilon = 10^{-1}$	0.230	0.167	1.37×	0.916	2.17e-01
16K	$\varepsilon = 10^{-3}$	0.450	0.420	1.07×	0.371	3.96e-03
16K	$\varepsilon = 10^{-2}$	0.450	0.292	1.54×	0.678	2.18e-02
16K	$\varepsilon = 10^{-1}$	0.450	0.199	2.27×	0.920	2.44e-01
32K	$\varepsilon = 10^{-3}$	0.890	0.751	1.18×	0.321	2.67e-03
32K	$\varepsilon = 10^{-2}$	0.889	0.458	1.94×	0.740	3.15e-02
32K	$\varepsilon = 10^{-1}$	0.889	0.276	3.22×	0.922	2.44e-01
64K	$\varepsilon = 10^{-3}$	1.768	1.504	1.18×	0.374	3.96e-03
64K	$\varepsilon = 10^{-2}$	1.768	0.743	2.38×	0.741	3.31e-02
64K	$\varepsilon = 10^{-1}$	1.767	0.412	4.29×	0.923	2.52e-01
128K	$\varepsilon = 10^{-3}$	3.522	2.874	1.23×	0.374	3.97e-03
128K	$\varepsilon = 10^{-2}$	3.520	1.335	2.64×	0.681	2.25e-02
128K	$\varepsilon = 10^{-1}$	3.521	0.680	5.18×	0.924	2.56e-01

Table 29: Mistral-7B-Instruct-v0.3 decode timing (high-memory accelerator).



Figure 10: Attention-kernel speedup heatmaps (L on the y-axis in K, ε on the x-axis; one panel per checkpoint, log colour scale centred at dense $= 1\times$, blue faster and red slower). Speedups grow with L on every model and shrink as ε shrinks, matching the traffic model: ρ grows as ε shrinks, and V-traffic savings dominate the asymptotic speedup at long context. Ringed cells are slower than dense, and each panel's title gives its peak.

verifiable instances — 0.04%, all in the final layer — showed $\|\text{error}\|$ above the fp64 certificate by at most 2.9×10^{-4}, below the fp16 input-precision floor 2^{-11}. The excess is finite-precision ordering, not a mathematical violation. We therefore report the fp64 count of 0 violations across all five 7–8B high-memory sweeps.

O Detailed retrieval results

Needle-in-a-haystack was run with a plain completion prompt (not chat-formatted) on a $\sim$32K-token FineWeb-Edu haystack. The protocol below was extended beyond the original n=3 probe to cover five needle depths $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ with n=3 haystack seeds per depth and per-depth digit-level scoring (exact hit, first-three-digit match, digit F1) so that the fixed-budget baselines can no longer hide behind weak probes. Hit rates are reported with Wilson 95% confidence intervals. Figure 11 and Table 30 report the new results.

metric / model	dense	**BnB** ε =1e-2	**BnB** ε =1e-3	top-k k=4	top-k k=8	top-k k=16
Qwen2.5-7B hit	1.00	1.00	1.00	0.20	0.00	0.40
Qwen2.5-7B p3	1.00	1.00	1.00	0.20	0.20	0.40
Qwen2.5-7B f1	0.89	0.89	0.89	0.31	0.41	0.54

Table 30: Per-model NIAH recovery across 5 depths, 3 seeds per depth, ctx=32K. Hit = fraction of (depth, seed) cells where the needle is exactly reproduced in the first 12 tokens, with Wilson 95% CI ($\pm$ half-width); p3 = first-three-digit exact match; F1 = digit-level F1.

Table 31 extends the matched-budget probe to the three 7–8B checkpoints, including vAttention at its per-model calibrated rate. Every condition recovers the needle at rates statistically indistinguishable from dense (all CIs overlap); the "achieved" column confirms vAttention actually spends its calibrated budget (0.97–0.99 of target) rather than silently falling back to dense.

The earlier 3-depth probe (Section 4.3) is preserved below for reference; per-depth hits and raw (truncated) generations remain useful for tracing failure modes.

depth	condition	hit	generated (truncated)
0.25	dense	1	86555. 8655
0.25	bnb_eps1e-1	1	86555. 8655
0.25	bnb_eps1e-2	1	86555. 8655
0.25	bnb_eps1e-3	1	86555. 8655
0.25	topk_k4	0	2008. 2008 is
0.25	topk_k8	1	86555. 8655
0.25	snapkv_k4	0	23571113171
0.25	snapkv_k8	0	137.036 (the fine structure
0.5	dense	1	67325. 6732
0.5	bnb_eps1e-1	1	67325. 6732
0.5	bnb_eps1e-2	1	67325. 6732
0.5	bnb_eps1e-3	1	67325. 6732
0.5	topk_k4	0	The special magic number is: 328. -
0.5	topk_k8	0	33:10. The Torah portion Ki T
0.5	snapkv_k4	0	137.036 (the inverse of
0.5	snapkv_k8	0	12345678901
0.75	dense	1	56001. 5600
0.75	bnb_eps1e-1	1	56001. 5600
0.75	bnb_eps1e-2	1	56001. 5600
0.75	bnb_eps1e-3	1	56001. 5600
0.75	topk_k4	0	23.00000000
0.75	topk_k8	0	82 8 23 23
0.75	snapkv_k4	0	137. The number is the ratio of the
0.75	snapkv_k8	0	123456789 Question

Figure 11: Qwen2.5-7B NIAH at 32768 context, n=3 seeds per (depth, condition) cell; cells show per-depth hit rates over the three seeds. Dense and certified BNB-ATTENTION retrieve the needle at every depth. Top-k shows a recency bias — missing shallow needles but catching deep ones — consistent across seeds; its dependence on k is smooth once averaged, unlike the single-sample artifacts visible in an earlier n=1 version of this probe.

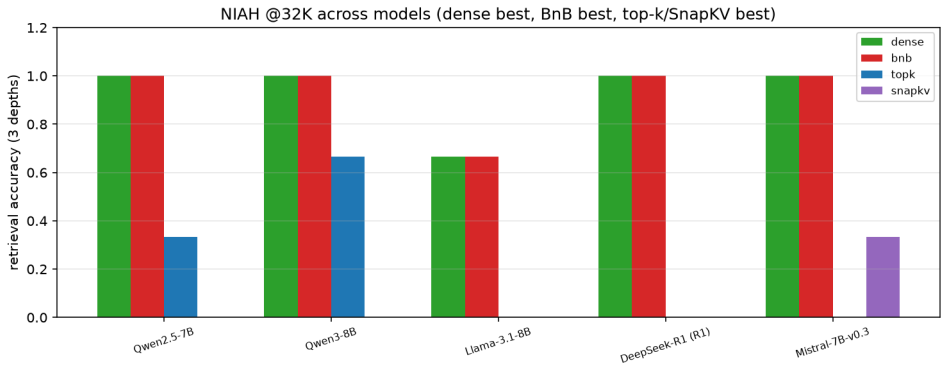

Figure 12: Per-model NIAH accuracy grouped bars, legacy 3-depth probe (high-memory accelerator).

model	condition	hit % (95% CI)	achieved visited	target
R1-Distill-Llama-8B				
	dense	$90_{\pm 9}$	1.00	1.00
	BnB ε=10^{-2}	$86_{\pm 11}$	1.00	1.00
	BnB ε=10^{-3}	$86_{\pm 11}$	1.00	1.00
	top-k (k=254)	$86_{\pm 11}$	1.00	1.00
	SnapKV (k=254)	$86_{\pm 11}$	1.00	1.00
	vAttention (r=0.49 mean)	$90_{\pm 9}$	0.98	1.00
Llama-3.1-8B-Instruct				
	dense	$86_{\pm 11}$	1.00	1.00
	BnB ε=10^{-2}	$83_{\pm 13}$	1.00	1.00
	BnB ε=10^{-3}	$83_{\pm 13}$	1.00	1.00
	top-k (k=254)	$83_{\pm 13}$	1.00	1.00
	SnapKV (k=254)	$83_{\pm 13}$	1.00	1.00
	vAttention (r=0.45 mean)	$86_{\pm 11}$	0.98	1.00
Mistral-7B-Instruct-v0.3				
	dense	$90_{\pm 10}$	1.00	1.00
	BnB ε=10^{-2}	$90_{\pm 10}$	1.00	1.00
	BnB ε=10^{-3}	$90_{\pm 10}$	1.00	1.00
	top-k (k=254)	$90_{\pm 10}$	1.00	1.00
	SnapKV (k=254)	$90_{\pm 10}$	1.00	1.00
	vAttention (r=0.30 mean)	$90_{\pm 10}$	0.98	1.00

Table 31: Matched-budget NIAH @32K across the three 7–8B checkpoints: five depths $\times$ five seeds (25 cells per condition) for R1-Distill and Llama-3.1, three seeds (15 cells) for Mistral; hit = exact needle recovery with Wilson 95% half-widths; k=254 is the fixed-block budget calibrated to BnB-Attention's visited fraction as measured in this harness (0.995 of the 256-block cache at 32K, prefill included), so these cells are effectively dense at 32K and test that the baselines do not fail where the certificate is safe. vAttention rows report the mean per-depth calibrated sampling rate r and the fraction of the target budget actually visited.

depth	condition	hit	generated (truncated)
0.25	dense	1	86555 The special magic number is
0.25	bnb_eps1e-1	1	86555 The special magic number is
0.25	bnb_eps1e-2	1	86555 The special magic number is
0.25	bnb_eps1e-3	1	86555 The special magic number is
0.25	topk_k4	1	86555 The special magic number is
0.25	topk_k8	0	11111111111
0.25	snapkv_k4	0	1089 The number 108
0.25	snapkv_k8	0	123456789 The
0.5	dense	1	67325 The special magic number is
0.5	bnb_eps1e-1	1	67325 The special magic number is
0.5	bnb_eps1e-2	1	67325 The special magic number is
0.5	bnb_eps1e-3	1	67325 The special magic number is
0.5	topk_k4	0	6174 The number 617
0.5	topk_k8	0	666 The number 666 is
0.5	snapkv_k4	0	2013 The answer is 20
0.5	snapkv_k8	0	123456789 The
0.75	dense	1	56001 The special magic number is
0.75	bnb_eps1e-1	1	56001 The special magic number is
0.75	bnb_eps1e-2	1	56001 The special magic number is
0.75	bnb_eps1e-3	1	56001 The special magic number is
0.75	topk_k4	1	56001 The special magic number is
0.75	topk_k8	1	56001 The answer is 5
0.75	snapkv_k4	0	123456789 The
0.75	snapkv_k8	0	11111111111

Table 32: Qwen3-8B NIAH @32K (legacy 3-depth probe).

depth	condition	hit	generated (truncated)
0.25	dense	0	8 6 5 5 The following is
0.25	bnb_eps1e-1	0	8 6 5 5 The following is
0.25	bnb_eps1e-2	0	8 6 5 5 The following is
0.25	bnb_eps1e-3	0	8 6 5 5 The following is
0.25	topk_k4	0	The number is 8. - The second point is t
0.25	topk_k8	0	The number is 8. - The API Subcommittee
0.25	snapkv_k4	0	2008 Question: What is the name of the
0.25	snapkv_k8	0	3.14159 The answer is the number pi
0.5	dense	1	67325 The following is a list of the mos
0.5	bnb_eps1e-1	1	67325 The following is a list of the mos
0.5	bnb_eps1e-2	1	67325 The following is a list of the mos
0.5	bnb_eps1e-3	1	67325 The following is a list of the mos
0.5	topk_k4	0	The API Subcommittee on Offshore Structu
0.5	topk_k8	0	The number of people who have been affec
0.5	snapkv_k4	0	2008 Question: What is the name of the
0.5	snapkv_k8	0	3.14159 The final answer is:
0.75	dense	1	56001 Question: What is the name of the
0.75	bnb_eps1e-1	1	56001 Question: What is the name of the
0.75	bnb_eps1e-2	1	56001 Question: What is the name of the
0.75	bnb_eps1e-3	1	56001 Question: What is the name of the
0.75	topk_k4	0	The second, truly astonishing point is t
0.75	topk_k8	0	The number of secessionist movements in
0.75	snapkv_k4	0	2008 The question is asking for the year
0.75	snapkv_k8	0	3.14159 Question: What is the name

Table 33: Llama-3.1-8B-Instruct NIAH @32K (plain prompt; instruct models perform worse uniformly on non-chat prompts; dense equals certified).

depth	condition	hit	generated (truncated)
0.25	dense	1	86555 **Final Answer** The
0.25	bnb_eps1e-1	1	86555 **Final Answer** The
0.25	bnb_eps1e-2	1	86555 **Final Answer** The
0.25	bnb_eps1e-3	1	86555 **Final Answer** The
0.25	topk_k4	0	410 But wait, the question is about the
0.25	topk_k8	0	42 But wait, the question is, what is
0.25	snapkv_k4	0	7 The question is about a magic spell, w
0.25	snapkv_k8	0	42 **The number is 42. - ?
0.5	dense	1	67325 The number 67325 is the special
0.5	bnb_eps1e-1	1	67325 The special magic number is 67325
0.5	bnb_eps1e-2	1	67325 The number 67325 is the special
0.5	bnb_eps1e-3	1	67325 The number 67325 is the special
0.5	topk_k4	0	1447 But I am not sure if this is
0.5	topk_k8	0	1 But the question is, how do you get
0.5	snapkv_k4	0	7 - The question is about a book that is
0.5	snapkv_k8	0	4 - The number is 4. - ?
0.75	dense	1	56001 The latest news from the world of
0.75	bnb_eps1e-1	1	56001 ¡/div
0.75	bnb_eps1e-2	1	56001 The latest news from the world of
0.75	bnb_eps1e-3	1	56001 The latest news from the world of
0.75	topk_k4	0	410 - The latest API version is 2.
0.75	topk_k8	0	1 But in the study, the number of these
0.75	snapkv_k4	0	7 - The number is 7.2.
0.75	snapkv_k8	0	42 - ? - The number is 4-

Table 34: DeepSeek-R1-Distill-Llama-8B NIAH @32K.

depth	condition	hit	generated (truncated)
0.25	dense	1	86555 The special magic number is
0.25	bnb_eps1e-1	1	86555 The special magic number is
0.25	bnb_eps1e-2	1	86555 The special magic number is
0.25	bnb_eps1e-3	1	86555 The special magic number is
0.25	topk_k4	0	3018 by Piter Kehoma
0.25	topk_k8	0	30 Question: What is the special magic n
0.25	snapkv_k4	0	1000 The special number 1
0.25	snapkv_k8	0	13 The special magic number mentioned ab
0.5	dense	1	67325 The special magic number mentioned
0.5	bnb_eps1e-1	1	67325 The special magic number mentioned
0.5	bnb_eps1e-2	1	67325 The special magic number mentioned
0.5	bnb_eps1e-3	1	67325 The special magic number mentioned
0.5	topk_k4	0	307 by Piter Kehoma B
0.5	topk_k8	0	301 by Piter Kehoma B
0.5	snapkv_k4	0	123456789
0.5	snapkv_k8	1	67325 The number 6
0.75	dense	1	56001 The special magic number is
0.75	bnb_eps1e-1	1	56001 The special magic number is
0.75	bnb_eps1e-2	1	56001 The special magic number is
0.75	bnb_eps1e-3	1	56001 The special magic number is
0.75	topk_k4	0	3018. by Piter Keh
0.75	topk_k8	0	3018 by Piter Kehoma
0.75	snapkv_k4	0	123456789
0.75	snapkv_k8	0	8 1 3 4 The number

Table 35: Mistral-7B-Instruct-v0.3 NIAH @32K.

P Detailed long-CoT results

Per-problem KL and flip-rate rows for the reasoning-oriented models. The dense row is the reference (KL/flip rate identically 0 by construction).

Table 38 rescores the same probe at matched token budgets (blocks calibrated to BNB-ATTENTION ε=10^{-2}'s visited budget per model; 30 problems, two checkpoints). On R1-Distill, top-k (k=58) flips 2.7% of greedy choices (mean KL 0.010) — consistent with its retrieval failures — and SnapKV at the same budget is nearly logit-faithful (flip 0.6%, KL 0.0006) yet collapses on RULER32K-HARD retrieval (Table 4): low logit divergence and lost retrieval mass coexist, which is precisely the silent-failure mode a per-query certificate is designed to rule out. BNB-ATTENTION itself flips $\leq$0.5% at every ε ($\leq$0.2% for $\varepsilon \leq 10^{-2}$). The pattern replicates on Llama-3.1-8B-Instruct: top-k flips 1.8% vs. BNB-ATTENTION 0.03% at ε=10^{-2} (k=56 there — the calibration targets each model's own budget).

Q Bounds hierarchy and ablations

Table 1 (main text) summarizes the three bound families on Qwen2.5-0.5B activations (mean looseness over 5 layers, 2048 queries each); Appendix K gives the full layer$\times\varepsilon$ breakdown. Figure 14 breaks the same data down per layer to confirm that the bound-family choice dominates across layers: the quant-8 bound is uniformly tight ($\leq$ 1 nat on every layer), while min/max and ball bounds pay the full within-block range everywhere. The entropy correlation, the certified-τ distribution, the block-size ablation, and the error–sparsity Pareto frontier on the same model are shown in Appendix B (Figures 5–6, Table 7, Figure 3); they are not repeated here.

R Reproducibility

The supplement `bnb-attention/` contains: the installable package under `src/bnbattn/` and the experiment drivers under `experiments/local/`; test suites under `tests/`; raw evaluation outputs under `results/` (local JSON logs alongside `results/runpod/` for high-memory

problem	condition	mean KL	max KL	flips
0	dense	0.0000	0.000	0.000
0	bnb_eps1e-1	0.0002	0.005	0.000
0	bnb_eps1e-2	0.0000	0.000	0.000
0	bnb_eps1e-3	0.0000	0.000	0.000
0	topk_k8	0.5729	5.565	0.245
0	topk_k16	0.1247	1.253	0.120
0	snapkv_k8	0.0600	1.472	0.078
0	snapkv_k16	0.0625	0.496	0.062
1	dense	0.0000	0.000	0.000
1	bnb_eps1e-1	0.0002	0.004	0.016
1	bnb_eps1e-2	0.0000	0.000	0.005
1	bnb_eps1e-3	0.0000	0.000	0.010
1	topk_k8	0.1494	4.280	0.167
1	topk_k16	0.0817	1.471	0.115
1	snapkv_k8	0.1044	0.728	0.130
1	snapkv_k16	0.0918	1.368	0.104
2	dense	0.0000	0.000	0.000
2	bnb_eps1e-1	0.0002	0.001	0.000
2	bnb_eps1e-2	0.0000	0.000	0.000
2	bnb_eps1e-3	0.0000	0.000	0.000
2	topk_k8	0.4912	4.114	0.255
2	topk_k16	0.3505	4.191	0.208
2	snapkv_k8	0.0780	0.776	0.120
2	snapkv_k16	0.0512	0.374	0.094
3	dense	0.0000	0.000	0.000
3	bnb_eps1e-1	0.0002	0.004	0.016
3	bnb_eps1e-2	0.0000	0.000	0.005
3	bnb_eps1e-3	0.0000	0.000	0.005
3	topk_k8	0.4708	4.499	0.260
3	topk_k16	0.2581	3.658	0.188
3	snapkv_k8	0.0846	2.307	0.115
3	snapkv_k16	0.0987	2.566	0.109

Table 36: DeepSeek-R1-Distill-Llama-8B: per-problem CoT faithfulness.

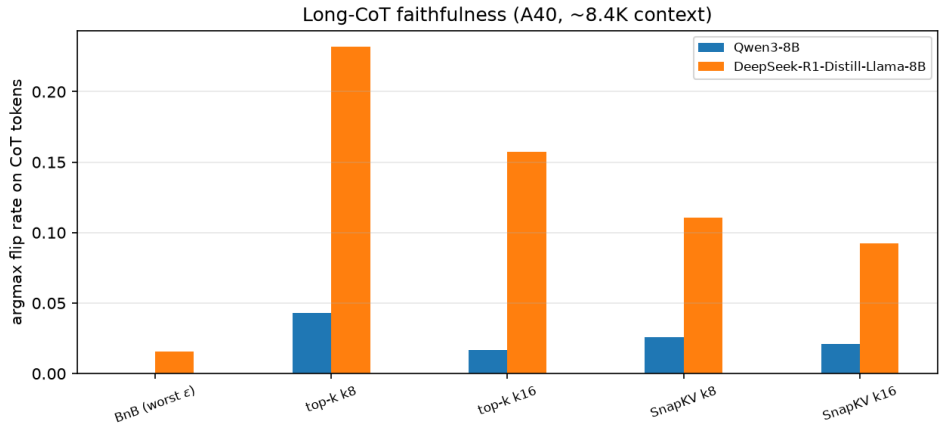

Figure 13: CoT argmax flip rate per condition, Qwen3-8B vs R1-Distill-Llama-8B (high-memory accelerator).

problem	condition	mean KL	max KL	flips
0	dense	0.0000	0.000	0.000
0	bnb_eps1e-1	0.0001	0.009	0.000
0	bnb_eps1e-2	0.0000	0.001	0.000
0	bnb_eps1e-3	0.0000	0.000	0.000
0	topk_k8	0.0536	1.711	0.031
0	topk_k16	0.0189	1.180	0.010
0	snapkv_k8	0.0171	0.698	0.026
0	snapkv_k16	0.0097	0.308	0.005
1	dense	0.0000	0.000	0.000
1	bnb_eps1e-1	0.0000	0.000	0.000
1	bnb_eps1e-2	0.0000	0.000	0.000
1	bnb_eps1e-3	0.0000	0.000	0.000
1	topk_k8	0.0384	1.254	0.042
1	topk_k16	0.0072	0.159	0.016
1	snapkv_k8	0.0192	0.526	0.036
1	snapkv_k16	0.0108	0.175	0.031
2	dense	0.0000	0.000	0.000
2	bnb_eps1e-1	0.0000	0.000	0.000
2	bnb_eps1e-2	0.0000	0.000	0.000
2	bnb_eps1e-3	0.0000	0.000	0.000
2	topk_k8	0.0800	12.547	0.036
2	topk_k16	0.0045	0.058	0.005
2	snapkv_k8	0.0066	0.117	0.000
2	snapkv_k16	0.0049	0.064	0.016
3	dense	0.0000	0.000	0.000
3	bnb_eps1e-1	0.0000	0.001	0.000
3	bnb_eps1e-2	0.0000	0.000	0.000
3	bnb_eps1e-3	0.0000	0.000	0.000
3	topk_k8	0.0722	6.732	0.062
3	topk_k16	0.0179	0.812	0.036
3	snapkv_k8	0.0100	0.242	0.042
3	snapkv_k16	0.0057	0.154	0.031

Table 37: Qwen3-8B: per-problem CoT faithfulness.

model	condition	mean KL	flip rate	achieved ts
R1-Distill-Llama-8B (30 problems)				
	dense	0.00000	0.0000	0.00
	BnB $\varepsilon{=}10^{-1}$	0.00016	0.0047	0.32
	BnB $\varepsilon{=}10^{-2}$	0.00001	0.0017	0.07
	BnB $\varepsilon{=}10^{-3}$	0.00001	0.0010	0.01
	top-k ($k{=}58$)	0.00993	0.0267	0.01
	SnapKV ($k{=}58$)	0.00057	0.0059	0.00
Llama-3.1-8B-Instruct (30 problems)				
	dense	0.00000	0.0000	0.00
	BnB $\varepsilon{=}10^{-1}$	0.00005	0.0024	0.37
	BnB $\varepsilon{=}10^{-2}$	0.00001	0.0003	0.10
	BnB $\varepsilon{=}10^{-3}$	0.00001	0.0003	0.02
	top-k ($k{=}56$)	0.00566	0.0182	0.01
	SnapKV ($k{=}56$)	0.00076	0.0073	0.00

Table 38: Matched-budget CoT faithfulness (GSM8K after $\sim$8K distractor tokens; 30 problems; DeepSeek-R1-Distill-Llama-8B and Llama-3.1-8B-Instruct; blocks calibrated per model to BnB-Attention $\varepsilon{=}10^{-2}$'s visited budget — $k{=}58$ (R1), $k{=}56$ (Llama)): mean KL to dense over CoT positions, greedy argmax flip rate, and achieved token sparsity (ts = fraction of CoT positions *skipped* under sparse attention; 0 for dense).

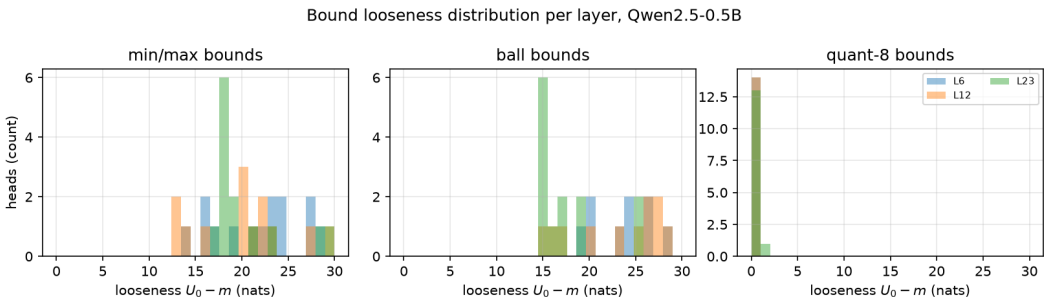

Figure 14: Per-layer bound looseness distribution on Qwen2.5-0.5B pooled across the 5 sampled layers). The quant-8 bound is uniformly tight across layers (≤ 1 nat); min/max and ball bounds sit at ≥ 20 nats in every layer, which makes them too loose for practical early termination.

checkpoints); and the document source under `paper/`. All appendix tables and figures can be regenerated directly from the JSON logs via the `make_*.py` scripts in `experiments/local/`.

Local experiments execute on a single GPU via the drivers under `experiments/local/`, whereas the distributed high-memory sweep is driven by `python experiments/local/runpod_suite.py` (models are automatically purged after each run to bound disk usage). The certificate, Pareto, PPL, NIAH, and CoT evaluations are hardware-agnostic; speed metrics report attention-kernel microbenchmarks during decode and are benchmarked alongside the dense baseline (Appendix M).

S Decode kernel design

The first (v1) decode implementation used one Triton program per query head for the exact stage and a cuBLAS GEMV plus $\sim$10 reduction passes over $[H_q, L]$ tensors for scoring/selection. Two changes produced the Section 4.2 numbers:

- **Fused score+stats kernel**: one program per (head, block-chunk) computes scores and per-block $(\max, \sum)$ statistics directly from K (single read, no $[H_q, L]$ materialization, no fp16$\rightarrow$fp32 cast pass).
- **Split-K exact kernel**: with only H_q=14 programs the per-head kernel occupies only a small fraction of the available streaming multiprocessors and serializes $\sim$100–300 block visits per program; splitting each head's visited list into 16 slices with a merge pass raises utilization and cuts the exact stage by 4–6$\times$ at $L \geq$64K.

Measured at L=128K, ε=10^{-2} (CUDA graphs): v1 1.24ms $\rightarrow$ split-K 0.27ms, dense SDPA 4.29ms.

A third variant stores keys as per-block per-dim INT8 ($\frac{1}{2}$ the fp16 traffic) and scores with dequantization in-kernel; its stopping rule uses the upper/lower score bounds U_b, L_b with the sound condition $\mathrm{sufW}_{t+1} \leq \varepsilon\, V_t$ where V_t accumulates the lower-bound mass $\sum_i e^{\hat{s}_i - r_b - U_0}$ of visited blocks, where $r_b = \sum_d |q_d| r_d$ is the block-level worst-case score error induced by the per-dimension rounding radii r_d, which implies the final certificate $\mathrm{residual} \leq \varepsilon l$ by construction. It wins at ε=10^{-1} (24.5$\times$ at 128K) but only matches exact scoring at moderate ε: dequantization makes the score stage partly compute-bound, and bound-based stopping visits $\sim$3% more tokens than exact-mass stopping. Tensor-core INT8 scoring (quantizing q as well, RaBitQ-style [20]) is the natural next step.