CERTBLOCK: CERTIFIED BLOCK SUMMARIES FOR BUDGET-CONSTRAINED LONG-CONTEXT SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Block-sparse attention serves a long context by committing a keep count per head and filling it with a selection rule, yet no step of that procedure bounds the attention mass the discarded blocks hold. Three challenges follow, each measured on 5 constructed workloads: an unobservable discarded mass, an unstable retained ordering as the workload is read, and incomparable per-head counts across differently concentrated heads. To address these challenges, we propose CertBlock, a block-sparse read-out that reports an upper bound on the mass it discards, comprising three components. Certified Block Summaries give every block a constant-size interval box and read from it an upper bound on the omitted softmax mass, carried as a certificate on the distance from dense attention. Certificate-Gated Selection admits and ranks blocks on that certified mass, a quantity that exists whether or not a block was selected. Global Budget Allocation spends one budget across heads and layers by the certified displacement each head incurs. The three components share one insight: the mass a truncated read-out omits can be bounded from a constant-size summary of each block, which turns the discarded portion into a computed term reported with every read-out. Because the rule computes its own certificate, a tolerance can refuse its selection. The certificate holds at all 1536 positions of each of two constructed workloads and at a sound rate of 1.000 on three held-out workloads under two attention geometries without retuning. At an identical committed block count, marginal allocation lowers the aggregated certified bound from 2.875 to 2.775 and from 4.921 to 4.852.

1 INTRODUCTION

Serving a long context under a fixed attention budget is now standard practice: the read-out attends over a chosen subset of key blocks and skips the rest (Yang et al., 2025; Zhang et al., 2025; Long et al., 2025; Yao et al., 2026). The procedure is fixed. A practitioner commits a keep count for each head, a selection rule fills that count with blocks (Ohayon et al., 2025; Liu et al., 2026), and the run is judged by the task loss it reaches. Every quantity that loop makes readable describes the blocks that were kept, and how much attention mass the discarded blocks held is never computed. The budget is spent against a number that no step of the procedure produces, which is the procedure the left panel of Figure 1 traces.

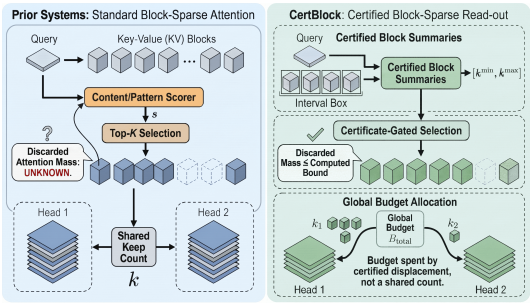

Figure 1: **Left:** a scored top-k under one shared keep count leaves the discarded mass unknown. **Right:** CertBlock bounds the discarded mass and spends one budget by certified displacement.

Three families of rule spend such a budget, each with a different answer to which blocks matter. The first scores a block from a summary of its own contents: content-based routing attends inside learned clusters (Roy et al., 2020), Block-Sparse FlashAttention (BSFA) keeps the blocks whose largest logit is highest (Ohayon et al., 2025), adaptive block sizing moves the partition with the content (Liu et al., 2026), and training-free selection drops blocks at inference (Zhang et al., 2025). The second learns the pattern: a trained selector picks the tokens to attend over (Wu et al., 2021),

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.

a gate stabilizes the sparse branch (Shen & Shen, 2026), and a block-sparse branch is combined with a linear one (Wang et al., 2026). The third carries state along the sequence: a key's future attention is estimated from unseen queries (Devoto et al., 2025), a key's norm alone decides what the cache keeps (Devoto et al., 2024), positions are pruned as the sequence advances (Long et al., 2025), a low-rank factorization replaces the removed attention (Li et al., 2025), and a serving stack fuses these forms behind one scheduler (Yang et al., 2025). Together these lines made a fixed budget cheap to spend and put block-sparse read-outs into standard stacks.

The three families share the shape of what they leave out: a content score is an ordering device whose scale carries no bound on the attention mass a block holds; the discarded portion of the softmax denominator is thus never computed. A learned score is updated only from the blocks it retained; hence the evidence that would argue for readmitting a block is destroyed by the act of not selecting it. A shared keep count is spent along an axis on which heads are not comparable, since the same count buys different amounts of retained mass in heads whose attention is differently concentrated. The consequence is measurable: at an identical committed block count, read-out displacements differ by 89.37 percent between two rules spending exactly the same budget, and no step of the standard procedure computes that difference. The question a budget should answer is therefore how much mass a retained set can prove it left behind.

To address these three challenges, we propose CertBlock, shown in the right panel of Figure 1, a block-sparse read-out that reports an upper bound on the mass it discards. CertBlock has three components: Certified Block Summaries, which give every block a constant-size interval box and read from it an upper bound on the mass that block would omit; Certificate-Gated Selection, which admits and ranks a block on that certified mass, a quantity that exists whether or not the block was selected, and caps the admitted set at the head's keep count; and Global Budget Allocation, which spends one budget across heads and layers by the certified displacement each head incurs. The key insight is that the mass a truncated read-out omits can be bounded from a constant-size summary of each block, which turns the discarded portion into a computed term reported with the read-out instead of an assumption. The bound is a property of the block summaries and of a selected set; hence any block-sparse rule can be read with it. Our contributions are as follows:

- We identify three challenges of block-sparse selection, each measured on 5 constructed workloads: an unobservable discarded mass, an unstable retained ordering, and incomparable per-head counts.
- We propose CertBlock, which bounds the omitted mass from a constant-size box per block, admits and ranks blocks on that certified mass, spends one budget across heads by certified displacement, and lets a tolerance refuse a selection whose ledger exceeds it.
- Our framework achieves a certificate holding at all 1536 positions per workload, a sound rate of 1.000 on three held-out workloads, and a marginal allocation lowering the aggregated bound below a uniform count, 2.775 against 2.875 and 4.852 against 4.921.

2 MOTIVATION

To understand what a block budget gives up when it is spent, we measure a plain top-k read-out over materialized block scores on 5 workloads and identify three challenges.

Challenge #1: Unobservable discarded mass. A block-sparse read-out commits a keep count and reports what it obtains, and what the discarded blocks were worth is never checked. The share of its own retained mass the rule can see is one at every keep count by construction. Figure 2 (left) reads, inside one decile of its unnormalized mass, how far the true retained share still spreads: it stands at 0.135 at a keep count of 8 blocks and never reaches zero over the sweep. Two queries the rule reads as identical therefore gave up different amounts, because a rule materializing only the scores of the blocks it keeps holds no quantity to check the rest with. *If two queries a rule reads as the same gave up different amounts of mass, what quantity should a keep count be checked against?*

Challenge #2: Unstable retained ordering. A selector learning from the blocks it keeps is argued safe because the blocks worth keeping are assumed to stay worth keeping, an assumption rarely tested against how much the true ordering moves. Figure 2 (middle) reads the top-k set of the true block mass over the early positions against the later ones at 22 window widths: the two sets overlap

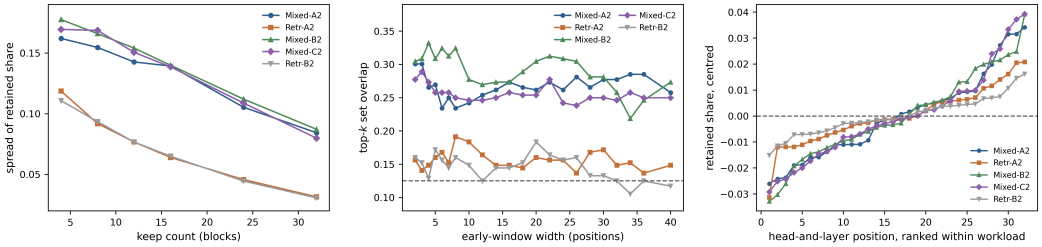

Figure 2: **Left:** Indistinguishable queries discard different amounts. **Middle:** The ordering does not persist. **Right:** Heads within a workload retain different amounts. Details in Appendix A.2.

by 0.209 on average and 0.117 at their lowest at the widest early window, against a chance overlap of 0.125. An ordering that reads no higher when more positions are read is not one a retained-only rule can lock onto and stay right. *If the ordering a retained-only rule would learn from stops holding as more of the workload is read, what should a block be ranked on instead?*

Challenge #3: Incomparable per-head counts. A budget is normally divided as one keep count per head, which takes what a keep buys to be comparable across heads, an assumption never read off the problem. We read the retained share head by head at one shared keep count. Figure 2 (right) ranks the heads inside each workload and subtracts the workload's own mean, leaving a spread of 5.684 points on average that every workload carries across its heads; on the absolute scale it runs from 0.528 to 0.889. A budget spent as a shared count is therefore spent along an axis on which the heads are not comparable, which defeats a uniform per-head count. *If one count buys different amounts in different heads, what should a budget be spent against?*

3 Problem Formulation

To address the three challenges of Section 2, we formulate budget-constrained long-context block selection as a constrained feasibility problem over a per-position selection rule σ:

$$\text{find } \sigma \in \mathcal{A} \quad \text{s.t.} \quad \mathcal{C}(\sigma) \leq B, \;\; \mathcal{D}(\sigma) \leq \epsilon, \tag{1}$$

where $\mathcal{A}$ is the class of admissible rules, each retaining at each query q a block set $S(q)$ ranked on a quantity that outlives its selection; $\mathcal{C}(\sigma)$ is the attention work the rule commits, under one budget B shared by all heads and layers; $\mathcal{D}(\sigma)$ is the certified displacement from the dense read-out of the same workload; and ϵ is the tolerance, an input the rule is given rather than a quantity read off it. The problem is feasibility rather than optimality: a rule either meets both constraints or is infeasible.

The constraint $\mathcal{D}(\sigma) \leq \epsilon$ answers Challenge #1: a ledger can be assembled only from a bound every retained set exposes. Membership in $\mathcal{A}$ answers Challenge #2: an admissible rule ranks a block on a quantity that outlives its own selection. The budget $\mathcal{C}(\sigma) \leq B$ answers Challenge #3: it is spent against incurred displacement rather than a per-head count.

4 Method

4.1 Overall Framework

To solve the feasibility problem of Eq. (1), we propose CertBlock, illustrated in Figure 3. Certified Block Summaries turn each block into a constant-size box and read from it an upper bound on the displacement a truncated read-out incurs, yielding the certificate $d(q, S)$. Certificate-Gated Selection admits and ranks blocks on their certified mass and caps them at the head's keep count, yielding the selected set $S(q)$. Global Budget Allocation spends the budget B across heads and layers by certified displacement, yielding the keep counts $\{k_{h,\ell}\}$. Details in Appendix A.3.

CertBlock rests on one interval box per block, yielding a bound on what truncation omits for every block, selected or not. The summaries combine that box with the retained mass into a certified mass and a certificate, the selector ranks on the mass, and the allocator spends by the certificate.

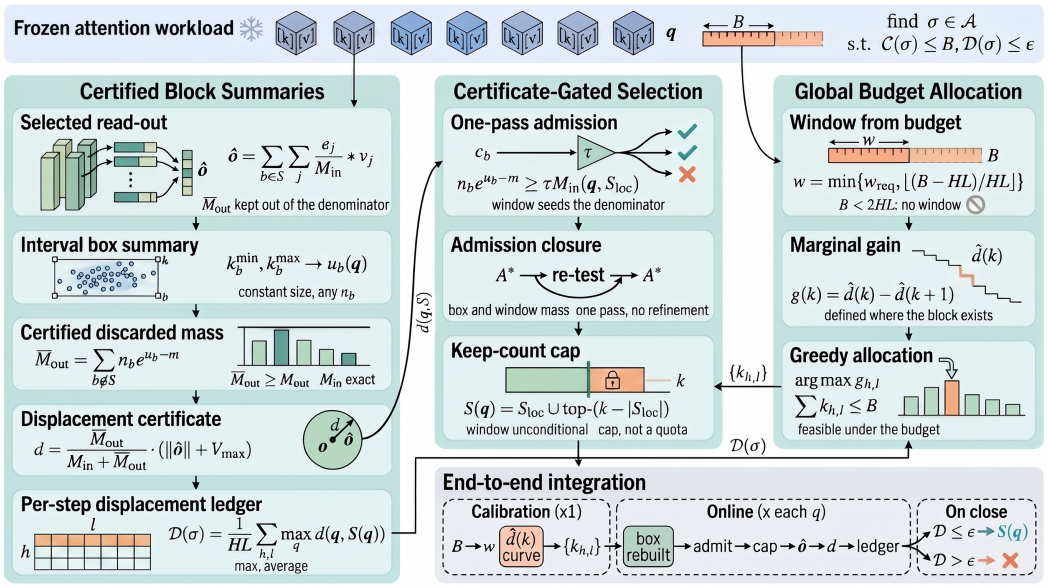

Figure 3: **Left:** Certified Block Summaries bound the omitted mass and displacement. **Middle:** Certificate-Gated Selection admits and caps by certified mass. **Right:** Global Budget Allocation spends B by marginal gain. **Bottom:** one calibration, one pass per query, an ϵ check on close.

4.2 Certified Block Summaries

To address Challenge #1 of Section 2, that the discarded mass is unobservable, we propose Certified Block Summaries, which bound the mass a read-out omits, in the four steps of the left panel of Figure 3: the box summary, the certified mass, the displacement certificate, and the ledger.

Selected read-out. Following the block-partition convention of Roy et al. (2020), the context is cut into key blocks, and each block b carries its keys $\{k_j\}$, values $\{v_j\}$ and size n_b. The read-out is the attention average restricted to the selected blocks, taken in the same forward pass,

$$\hat{o}\big(S(q)\big) = \sum_{b \in S(q)} \sum_{j \in b} \frac{\exp\big((q^\top k_j)/\sqrt{d_\mathrm{h}} - m(q)\big)}{M_\mathrm{in}(q, S(q))}\, v_j, \tag{2}$$

where $\hat{o}(S(q))$ is the selected read-out, k_j and v_j are the keys and values of block b, d_h is the head dimension, $m(q)$ is the per-query shift of Eq. (4), and M_in is the exactly retained mass. The certified discarded mass $\bar{M}_\mathrm{out}$ is an upper bound and not a measurement; it thus stays out of the denominator, where it would rescale the read-out by the looseness of the bound; it enters Eq. (5) alone.

Interval box summary. We compute, from the block's current keys at every step, the coordinatewise box $k_b^{\min}, k_b^{\max}$ and read off an upper bound on every scaled logit the block can produce for the query,

$$k_b^{\min} = \min_{j \in b} k_j, \quad k_b^{\max} = \max_{j \in b} k_j, \quad u_b(q) = \frac{1}{\sqrt{d_\mathrm{h}}} \sum_i \max\{q_i\, k_{b,i}^{\min},\ q_i\, k_{b,i}^{\max}\}, \tag{3}$$

where the minimum and maximum are coordinatewise over the block's keys, q_i is the i-th coordinate of the query, and $u_b(q)$ is the interval-arithmetic upper bound on the scaled logit.

Certified discarded mass. From these bounds we compute, relative to the per-query shift $m(q)$, an upper bound on the softmax mass any discarded set omits and the exactly retained mass inside the selected set,

$$m(q) = \max_b u_b(q),\ \bar{M}_\mathrm{out}(q, S) = \sum_{b \notin S} n_b\, e^{\,u_b(q) - m(q)},\ M_\mathrm{in}(q, S) = \sum_{b \in S} \sum_{j \in b} e^{\,(q^\top k_j)/\sqrt{d_\mathrm{h}} - m(q)}, \tag{4}$$

where $m(q)$ is the per-query shift that keeps the exponentials bounded, $\bar{M}_\mathrm{out}$ is the discarded-mass upper bound and M_in is the exactly computed retained mass. The block size n_b multiplies the box exponential because every key in the block is bounded by the same $u_b(q)$.

Displacement certificate. We then derive the displacement of one read-out under the selected set by passing the discarded-mass bound through the largest value norm in the discarded blocks,

$$d(q,S) = \frac{\bar{M}_{\mathrm{out}}(q,S)}{M_{\mathrm{in}}(q,S) + \bar{M}_{\mathrm{out}}(q,S)} \Big(\|\hat{o}(S)\| + V_{\max} \Big), \quad V_{\max} = \max_{b \notin S} V_b, \quad V_b = \max_{j \in b} \|v_j\|, \tag{5}$$

where V_b is the value radius of block b, $V_{\max}$ is the largest value radius over the discarded blocks, taken as zero when no block is discarded, in which case $\bar{M}_{\mathrm{out}} = 0$ and $d(q,S) = 0$, and $d(q,S)$ upper-bounds the displacement $\|\hat{o}(S) - o\|$ from the dense read-out o. The two read-outs normalize by different masses, and writing out the difference fixes the form of the bound:

$$\hat{o}(S) - o = \frac{M_{\mathrm{out}}(q,S)}{M_{\mathrm{in}}(q,S) + M_{\mathrm{out}}(q,S)} \Big(\hat{o}(S) - \bar{v}_{\mathrm{out}} \Big), \quad \bar{v}_{\mathrm{out}} = \frac{1}{M_{\mathrm{out}}} \sum_{b \notin S} \sum_{j \in b} e_j v_j, \tag{6}$$

with M_{out} the mass actually omitted, $e_j = e^{\,(q^\top k_j)/\sqrt{d_{\mathrm{h}}} - m(q)}$ the shifted weight of key j, d_{h} the head dimension, and $\bar{v}_{\mathrm{out}}$ the mass-weighted mean of the discarded values. The identity of Eq. (6) is exact. Both terms inside the bracket are then bounded, the first by the read-out's own norm and the second by $V_{\max} \geq \|\bar{v}_{\mathrm{out}}\|$, which yields the $\|\hat{o}(S)\| + V_{\max}$ factor of Eq. (5); a bound written on $V_{\max}$ alone would drop the first term. The box of Eq. (3) contains the block's keys at every step; hence the $\bar{M}_{\mathrm{out}}$ of Eq. (4) never underestimates M_{out}, and since $x \mapsto x/(M_{\mathrm{in}} + x)$ increases in x, substituting $\bar{M}_{\mathrm{out}}$ for M_{out} can only raise the fraction. The bound therefore holds at every query, and its slack has two sources: that substitution and the two norm bounds inside the bracket.

Per-step displacement ledger. The ledger takes, at each head and layer, the worst such bound over that head's reference positions and averages the result,

$$\mathcal{D}(\sigma) = \frac{1}{HL} \sum_{h,\ell} \max_{q \in \mathcal{Q}_{h,\ell}} d\big(q, S(q)\big), \tag{7}$$

where H and L count the heads and layers and $\mathcal{Q}_{h,\ell}$ collects the reference positions of head h at layer ℓ. The component thus produces, per query and per head, a forward pointwise number that upper-bounds the displacement of the truncated read-out, and aggregates those numbers into the ledger the deviation constraint of Eq. (1) is checked on.

4.3 Certificate-Gated Selection

To address Challenge #2, that the retained ordering is unstable as the workload is read, we propose Certificate-Gated Selection, which ranks a block on a quantity that exists while it is unselected, in three steps, the middle panel of Figure 3: one-pass admission, its closure, and a keep-count cap.

One-pass admission. We select the admitted set in a single pass as the mandatory local window together with every block whose certified mass clears a threshold fixed as a fraction of the window's exactly retained mass,

$$A^\star(q) = S_{\mathrm{loc}}(q) \cup \Big\{ b : n_b \, e^{\,u_b(q) - m(q)} \geq \tau \, M_{\mathrm{in}}\big(q, S_{\mathrm{loc}}(q)\big) \Big\}, \tag{8}$$

where $S_{\mathrm{loc}}(q)$ is the local window every position keeps unconditionally, τ is the admission threshold, and $M_{\mathrm{in}}(q, S_{\mathrm{loc}}(q))$ is the exactly retained mass of the window that seeds the denominator. The window makes the denominator a computed positive number; hence the test is well posed at the first query of a sequence.

Admission closure. The admission test restricts each block to the current box summaries and the window mass; hence re-applying it to the admitted set returns that set unchanged,

$$S_{\mathrm{loc}}(q) \cup \Big\{ b \in A^\star(q) : n_b \, e^{\,u_b(q) - m(q)} \geq \tau \, M_{\mathrm{in}}\big(q, S_{\mathrm{loc}}(q)\big) \Big\} = A^\star(q), \tag{9}$$

where the left side keeps the window and re-tests every other admitted block against the same threshold. The window enters both sides because Eq. (8) keeps it whether or not it clears the threshold.

Keep-count cap. We then cap the admitted set at the head's keep count, ranking by certified mass while retaining the window unconditionally,

$$S(q) = S_{\mathrm{loc}}(q) \cup \underset{k_{h,\ell} - |S_{\mathrm{loc}}(q)|}{\text{top-}} \Big\{ n_b \, e^{\,u_b(q) - m(q)} : b \in A^\star(q) \setminus S_{\mathrm{loc}}(q) \Big\}, \tag{10}$$

where $k_{h,\ell}$ is the keep count committed at head h of layer ℓ and $S(q)$ is the selected set after the cap. The count is a cap: when Eq. (8) returns fewer blocks than it allows, $|S(q)|$ falls below $k_{h,\ell}$; accordingly, the committed count and the realized retention are reported as separate columns. A block is ranked on its certified mass and not on whether it was previously kept. The component therefore hands the read-out a selected set $S(q)$ whose every ranked member earned its place on a quantity that persists across steps, which renders σ admissible under Eq. (1).

4.4 Global Budget Allocation

To address Challenge #3, that one keep count buys different amounts in different heads, we propose Global Budget Allocation, which spends B by the certified displacement each head incurs, in three steps, the right panel of Figure 3: a window from the budget, a marginal gain, and a greedy sweep.

Window from budget. We set the window length from the budget before the sweep is run, which is well defined only when the budget leaves one window block and one ranked block per head,

$$w = \min\Big\{w_{\mathrm{req}},\ \Big\lfloor \frac{B - HL}{HL} \Big\rfloor\Big\}, \qquad B \geq 2HL, \tag{11}$$

where w_{req} is the requested window length, the floor term is the longest window the budget can carry, and H and L count the heads and layers. The condition is $B \geq 2HL$ rather than $B \geq HL$: at $B = HL$ the floor term is zero, the local set is empty, and the denominator of Eq. (8) is taken on an empty set. Taking the floor term as the window rather than as a ceiling on it would leave each head one block for Eq. (8) to Eq. (10) to place; the requested length is therefore used wherever the budget can carry it. Below $B = 2HL$ the construction refuses to return a window length.

Marginal gain. We compute, for each head and layer, the reduction in its certified bound from one more retained block,

$$g_{h,\ell}(k) = \hat{d}_{h,\ell}(k) - \hat{d}_{h,\ell}(k+1), \quad \hat{d}_{h,\ell}(k) = \max_{q \in \mathcal{Q}_{h,\ell}} d\big(q, S(q)\big)\big|_{k_{h,\ell}=k}, \tag{12}$$

where $\hat{d}_{h,\ell}(k)$ is the bound of Eq. (5) at keep count k maximized over the reference positions $\mathcal{Q}_{h,\ell}$, and $g_{h,\ell}(k)$ is the marginal gain, defined only where that block exists.

Greedy allocation. We then solve the allocation by a greedy sweep that repeatedly gives the next block to the head with the largest current marginal gain, until the budget is exhausted,

$$(h^\star, \ell^\star) = \arg\max_{h,\ell}\ g_{h,\ell}(k_{h,\ell}), \qquad \{k_{h,\ell}\} : \sum_{h,\ell} k_{h,\ell} \leq B, \tag{13}$$

where $(h^\star, \ell^\star)$ is the head and layer the sweep gives the next block to and $\{k_{h,\ell}\}$ is the resulting allocation under the single budget $\mathcal{C}(\sigma) = \sum_{h,\ell} k_{h,\ell} \leq B$. The marginal-gain sequence need not be non-increasing: the allocation the sweep returns is feasible under the budget rather than least-bound. A ledger above ϵ returns a refusal and no selected sets, which is the only place the tolerance enters. The component turns the single budget into a per-head keep count $\{k_{h,\ell}\}$ set by incurred displacement; the committed work $\mathcal{C}(\sigma)$ thus respects B while heads needing more blocks receive them. The table in Appendix A.15 places this allocation against a uniform per-head keep count.

End-to-end integration. The three components run as one construction in two phases, shown in the bottom panel of Figure 3. Calibration runs once: it sets the window by Eq. (11), builds the bound curve of Eq. (12) over every keep count, and fixes $\{k_{h,\ell}\}$ by the sweep of Eq. (13). The online phase runs at each query: the box summaries of Eq. (3) are rebuilt, the admission and cap of Eq. (8) to (10) produce the selected set, the read-out of Eq. (2) is taken over it, and the certificate of Eq. (5) is recorded into the ledger checked against the tolerance on close. Details in Appendix A.4.

5 Experiments

5.1 Implementation Details

Workloads. Every reading is computed on a workload, a key, value and query tensor built rather than recorded, with its mass over blocks concentrated, not flat; rules are thus compared on a common tensor; the benchmark, held-out and motivation sets are disjoint. Details in Appendix A.5.

Table 1: Main benchmark under a shared block budget. Refusable: the rule computes the bound itself; hence a tolerance can refuse its selection.

Workload	Rule	Agr.	Displ.	Cert.	Refusable
Retr-A	BSFA	1.000	0.056	1.268	no
	AB-Sparse	0.413	0.640	6.198	no
	Hybrid	1.000	0.072	1.560	no
	CertBlock	**1.000**	**0.068**	**1.455**	**yes**
Mixed-A	BSFA	0.924	0.103	3.198	no
	AB-Sparse	0.474	0.338	6.252	no
	Hybrid	0.906	0.131	3.743	no
	CertBlock	**0.917**	**0.137**	**3.578**	**yes**

Baselines. Three block-sparse rules are read against CertBlock at the same committed block count. BSFA keeps the blocks whose largest logit is highest at a uniform keep count (Ohayon et al., 2025); AB-Sparse scores the blocks it has materialized per head (Liu et al., 2026); and Hybrid Sparse Attention (Hybrid) spends a per-head budget alongside a local window (Yao et al., 2026). Each ranks on the exact block quantity its description names, read off the workload. Details in Appendix A.6.

Evaluation Metrics. Dense agreement is the share of positions at which the truncated read-out selects the same value centroid as the dense read-out (↑). Read-out displacement is the distance from that dense read-out (↓), and the certificate of Eq. (5) bounds it (↓); the sound rate is the share of positions at which the certificate stands at or above the displacement it covers (↑). Realized retention and the ledger of Eq. (7) report what the budget bought. Details in Appendix A.7.

Hyperparameters. A workload carries 8 heads over 4 layers, 64 blocks of 64 keys, and 48 query positions per head. All rules share a retention budget of 0.375, over the 64×32 block slots; CertBlock runs a requested window of 8 blocks and an admission threshold $\tau = 0.08$. The tolerance is $\epsilon = 8.0$. Details in Appendix A.8.

Software. Every table, number and data figure is produced by one entry point, `code/run.py`, at pinned NumPy and Matplotlib versions. The only random draws are the seeded workload constructions; a second run thus reproduces the first. Details in Appendix A.9.

Protocol. Every rule is read at the same committed block count, such that two rules differ only in where the budget went. Calibration runs once over all 1536 query positions of a workload; each ablation holds the budget fixed and replaces one quantity. Details in Appendix A.10.

5.2 Main Results

Shared-budget benchmark. Table 1 places the four rules at the same committed block count. The certified read-out carries a certificate of 1.455 and 3.578, against 1.268 and 3.198 for BSFA, and the realized displacement stayed under the bound at every position, a sound rate of 1.000 and 1.000. On displacement the certified rank reads 0.068 against 0.056 for the strongest baseline and 0.137 against 0.103 on the other. Agreement separates BSFA, Hybrid and CertBlock far less: the certified rank reads 1.000 on Retr-A, which the strongest baseline also attains, and 0.917 against 0.924 on the other, a shortfall of 0.78 percent. What the last column separates is availability: the certified rule alone computes its own bound and alone can be refused by a tolerance. Details in Appendix A.11.

Window at the budget bound. Table 2 reports one arm that holds the rule fixed and pushes the window to the bound of Eq. (11), 23 blocks at this budget, which leaves the ranking of Eq. (10) one block per head to place. Displacement rises on both workloads, 0.134 and 0.243 against 0.068 and 0.137. Agreement holds at 1.000 on Retr-A and falls from 0.917 to 0.840 on Mixed-A; the agreement cost of the wider window thus falls on the harder of the two workloads. As run, the certified ranking fills each head's keep count beyond its 8 window blocks; at the bound it places one block, and a window kept whether or not its blocks clear the threshold becomes nearly the whole selection, a recency rule. The lower displacement of the certified run is thus the ranking's contribution. Details in Appendix A.12.

Table 2: The window as run against the window at its bound.

Arm	w	Agr.	Displ.
Retr-A			
as run	8	1.000	0.068
at the bound	23	1.000	0.134
Mixed-A			
as run	8	0.917	0.137
at the bound	23	0.840	0.243

Table 3: Held-out workloads under two attention geometries. None of these workloads appears in the main table, and the rule is not retuned on any of them.

Workload	Geometry	Dense agr.	Displ. mean	Read-out ratio	Ledger $\mathcal{D}$
Mixed-B	8 heads, $d_{\mathrm{h}} = 64$	0.915	0.133	1.149	4.78
	4 heads, $d_{\mathrm{h}} = 32$	0.953	0.064	1.065	3.10
Mixed-C	8 heads, $d_{\mathrm{h}} = 64$	0.927	0.132	1.158	4.82
	4 heads, $d_{\mathrm{h}} = 32$	0.969	0.065	1.086	2.82
Retr-B	8 heads, $d_{\mathrm{h}} = 64$	1.000	0.065	1.061	2.71
	4 heads, $d_{\mathrm{h}} = 32$	1.000	0.011	1.010	0.58

Held-out workloads. Table 3 evaluates Mixed-B, Mixed-C and Retr-B under two attention geometries. The certificate holds on all six workload-geometry cells, at a sound rate of 1.000, as the box of Eq. (3) contains its block's keys on any workload, and the read-out norm ratio stays between 1.010 and 1.158, the same order as on the main workloads. At the main 8-head geometry each held-out workload tracks its family's main one in displacement, 0.133 and 0.132 against 0.137 on Mixed-A and 0.065 against 0.068 on Retr-A; the construction transfers. Details in Appendix A.13.

5.3 Ablation Study and Analysis

Certified mass. Table 6 reads each ablation against the condition it was built to meet, at one committed block count. Replacing the certified mass by the relevance of the block's mean key moves displacement by 0.00002 and 0.00018, both inside the 0.001 indistinguishability margin; hence this axis does not separate the two rankings. The arm keeps the window, the admission test, the keep counts and the read-out of the certified run. The ranking answers Challenge #2 through a quantity every block carries while unselected, which the mean-key score also is; a router that updates its score only on retained blocks moves displacement by 0.5009 and 0.1797. Details in Appendix A.14.

Marginal allocation. Table 4 compares marginal-certified against uniform per-head allocation at three budgets. The aggregated certified bound is lower under the marginal sweep on both workloads, at the operating point 2.775 against 2.875 and 4.852 against 4.921, and the bound columns keep that ordering at each budget. At the operating point the sweep spreads keep counts over 15 to 36 and 11 to 35 blocks per head against a uniform 24, the per-head difference of Challenge #3. Agreement with the dense read-out is lower in 3 of the 6 budget cells, the widest being the tightest cell of Mixed-A, 0.551 against 0.811. Both arms commit the same block count; the bound columns are thus read at equal cost. Details in Appendix A.15.

Table 4: Certified bound by allocation.

Retr-A			Mixed-A		
Budget	Marginal	Uniform	Budget	Marginal	Uniform
0.05	6.378	6.408	0.05	9.045	9.220
0.10	5.986	6.024	0.10	8.582	8.696
0.25	4.124	4.212	0.25	6.354	6.437

Admission threshold. Table 5 sweeps Eq. (8) over seven values of τ at a uniform keep count. Up to $\tau = 0.020$ almost every block is admitted and the reading does not move. At $\tau = 0.08$ the admitted share falls from 1.000 to 0.702 and from 1.000 to 0.713, and displacement rises by 0.00042 and 0.00078. Displacement rises once a head admits fewer blocks than its keep count. Eq. (9) is rechecked at every query of the main run and holds at all of them; one admission pass thus suffices. Details in Appendix A.16.

Table 5: Four of the seven thresholds swept.

	Retr-A		Mixed-A	
τ	Admitted	Displ.	Admitted	Displ.
0.000	1.000	0.06467	1.000	0.11942
0.020	0.957	0.06467	0.978	0.11942
0.080	0.702	0.06509	0.713	0.12019
0.320	0.294	0.09548	0.246	0.22134

Persistent ranking quantity. The figure in Appendix A.17 contrasts the certified rank with a router ranking on a score carried forward from the blocks it has already read. The router moves the never-selected fraction by 0.2563 and 0.3115 and the displacement by 0.5009 and 0.1797: the router never revisits dropped blocks, while the certified mass is recomputed for every block per query. On the first-to-last Jaccard overlap the certified rule sits between the two baselines: its first-to-last overlap reads 0.430 and 0.428, against 0.473 and 0.474 for Hybrid and 0.301 and 0.302 for BSFA.

Certificate soundness. The table in Appendix A.18 validates the two inequalities the certificate rests on at 5 retention levels, on 384 of 1536 positions. At every row the certified discarded mass

Table 6: Each ablation against the component it replaces.

Component	Reading	Workload	Rule	Ablation	Difference
Selection (mean key)	read-out displacement	Retr-A	0.0681	0.0681	0.0000
	read-out displacement	Mixed-A	0.1368	0.1366	-0.0002
Selection (router)	never-selected fraction	Retr-A	0.003	0.259	0.256
	first-to-last Jaccard	Retr-A	0.430	0.520	0.090
	read-out displacement	Retr-A	0.0681	0.5690	0.5009
	never-selected fraction	Mixed-A	0.009	0.321	0.312
	first-to-last Jaccard	Mixed-A	0.428	0.567	0.139
	read-out displacement	Mixed-A	0.1368	0.3165	0.1797
Budget allocation	aggregated certified bound	Retr-A	2.775	2.875	0.100
	aggregated certified bound	Mixed-A	4.852	4.921	0.069

stands above the mass actually discarded, and the certificate stands at or above the realized displacement at a rate of 1.000. On the main run it fell below at 0 of 1536 positions per workload.

Bound tightness. The table in Appendix A.19 quantifies the slack, the certificate minus the realized displacement. It grows from 2.000 to 3.467 as the within-block key spread widens from 0.020 to 0.600. Two sources produce it: the certified discarded mass exceeds the true one by a factor of 2.046 to 2.477, and the norm bounds of Eq. (5) replace a distance by a sum of two radii.

Per-step cost. The table in Appendix A.20 accounts for the box rebuild and the admission pass. Amortized over the queries of a step the certified read-out spends 0.409 of a dense pass against the baselines' 0.375; charged once per query it spends 1.389. The rebuild is linear in context length times head dimension, and the removed attention is linear in the discarded blocks: the construction adds a term growing with the context and removes one growing with what the budget discards.

6 Related Work

Block-sparse and content-routed attention. This line builds the block partition and the score that ranks it: content-based routing attends inside learned clusters (Roy et al., 2020), block-max pruning ranks a block by its largest logit (Ohayon et al., 2025), adaptive block sizing lets the partition move with the content (Liu et al., 2026), and a trained selector picks the tokens to attend over (Wu et al., 2021). Each ranks on a quantity whose scale says nothing about the mass the unselected blocks hold. CertBlock ranks instead on a quantity that upper-bounds that mass.

Training-free selection and cache compression. This line works under our constraint: a fixed model, a fixed budget, a fixed context. Training-free attention drops blocks at inference (Zhang et al., 2025), dynamic pruning drops positions as the sequence runs (Long et al., 2025), and cache compression evicts on predicted attention (Devoto et al., 2025) or on a key's norm alone (Devoto et al., 2024). What the discarded portion was worth is never computed: a rule that dropped nothing and one that dropped a third are compared only to each other; Eq. (5) puts them on one scale.

Budget allocation under attention concentration. This line splits one budget rather than making one selection: importance-driven allocation spends it where it is needed (Zhang et al., 2023a), and a hybrid rule spends a per-head budget beside a local window (Yao et al., 2026). What makes a uniform count the wrong unit is where the mass sits: the first positions absorb a large share of it, a concentration now understood as structural (Qiu et al., 2026; Son et al., 2024). CertBlock spends its keep counts against the certified displacement each head incurs.

7 Conclusion

This paper asked block selection under a budget to carry a number it does not produce: the mass a truncated read-out omits. CertBlock answers with three components: Certified Block Summaries bound that mass per block, Certificate-Gated Selection admits and ranks blocks on it, and Global Budget Allocation spends one budget by certified displacement. A tolerance can refuse a selection, the certificate is sound on every held-out cell, and marginal allocation lowers the aggregated bound at every tested budget. A block budget can therefore be spent against a bound the rule computes.

REFERENCES

Iz Beltagy, Matthew E. Peters, Arman Cohan. Longformer: The Long-Document Transformer. arXiv preprint, 2020.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, Wen Xiao. PyramidKV: Dynamic KV Cache Compression based on Pyramidal Information Funneling. arXiv preprint, 2025.

Alessio Devoto, Yu Zhao, Simone Scardapane, Pasquale Minervini. A Simple and Effective L_2 Norm-Based Strategy for KV Cache Compression. arXiv preprint, 2024.

Alessio Devoto, Maximilian Jeblick, Simon Jégou. Expected Attention: KV Cache Compression by Estimating Attention from Future Queries Distribution. arXiv preprint, 2025.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, S Kevin Zhou. Ada-KV: Optimizing KV Cache Eviction by Adaptive Budget Allocation for Efficient LLM Inference. In The Thirty-ninth Annual Conference on Neural Information Processing Systems, 2025.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, Jianfeng Gao. Model Tells You What to Discard: Adaptive KV Cache Compression for LLMs. In The Twelfth International Conference on Learning Representations, 2024.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, Lili Qiu. MInference 1.0: accelerating pre-filling for long-context LLMs via dynamic sparse attention. In Proceedings of the 38th International Conference on Neural Information Processing Systems, 2024.

Changqun Li, Chaofan Ding, Kexin Luan, Xinhan Di. Low-Rank Adaptation with Task-Relevant Feature Enhancement for Fine-tuning Language Models. arXiv preprint, 2024.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, Deming Chen. SnapKV: LLM knows what you are looking for before generation. In Proceedings of the 38th International Conference on Neural Information Processing Systems, 2024.

Tenghui Li, Guoxu Zhou, Xuyang Zhao, Yuning Qiu, Qibin Zhao. Efficient Low Rank Attention for Long-Context Inference in Large Language Models. arXiv preprint, 2025.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, Anshumali Shrivastava. Scissorhands: exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In Proceedings of the 37th International Conference on Neural Information Processing Systems, 2023.

Di Liu, Ruitian Wang, Chen Chen, Mingliang Gong, Yongjie Yuan, Han Zhao, Yu Feng, Quan Chen, Minyi Guo. AB-Sparse: Sparse Attention with Adaptive Block Size for Accurate and Efficient Long-Context Inference. arXiv preprint, 2026.

Lingkun Long, Rubing Yang, Yushi Huang, Desheng Hui, Ao Zhou, Jianlei Yang. SlimInfer: Accelerating Long-Context LLM Inference via Dynamic Token Pruning. arXiv preprint, 2025.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, Zhiqi Huang, Huan Yuan, Suting Xu, Xinran Xu, Guokun Lai, Yanru Chen, Huabin Zheng, Junjie Yan, Jianlin Su, Yuxin Wu, Yutao Zhang, Zhilin Yang, Xinyu Zhou, Mingxing Zhang, Jiezhong Qiu. MoBA: Mixture of Block Attention for Long-Context LLMs. In The Thirty-ninth Annual Conference on Neural Information Processing Systems, 2025.

Daniel Ohayon, Itay Lamprecht, Itay Hubara, Israel Cohen, Daniel Soudry, Noam Elata. Block Sparse Flash Attention. arXiv preprint, 2025.

Zihan Qiu, Zeyu Huang, Kaiyue Wen, Peng Jin, Bo Zheng, Yuxin Zhou, Haofeng Huang, Zekun Wang, Xiao Li, Huaqing Zhang, Yang Xu, Haoran Lian, Siqi Zhang, Rui Men, Jianwei Zhang, Ivan Titov, Dayiheng Liu, Jingren Zhou, Junyang Lin. A Unified View of Attention and Residual Sinks: Outlier-Driven Rescaling is Essential for Transformer Training. arXiv preprint, 2026.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, David Grangier. Efficient Content-Based Sparse Attention with Routing Transformers. arXiv preprint, 2020.

Alfred Shen, Aaron Shen. Gated Sparse Attention: Combining Computational Efficiency with Training Stability for Long-Context Language Models. arXiv preprint, 2026.

Seungwoo Son, Wonpyo Park, Woohyun Han, Kyuyeun Kim, Jaeho Lee. Prefixing Attention Sinks can Mitigate Activation Outliers for Large Language Model Quantization. arXiv preprint, 2024.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, Song Han. QUEST: query-aware sparsity for efficient long-context LLM inference. In Proceedings of the 41st International Conference on Machine Learning, 2024.

Bailin Wang, Dan Friedman, Tao Lei, Chong Wang. SPLA: Block Sparse Plus Linear Attention for Long Context Modeling. arXiv preprint, 2026.

Chuhan Wu, Fangzhao Wu, Tao Qi, Binxing Jiao, Daxin Jiang, Yongfeng Huang, Xing Xie. Smart Bird: Learnable Sparse Attention for Efficient and Effective Transformer. arXiv preprint, 2021.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In International Conference on Learning Representations, pp. 21875–21895, 2024.

Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, Maosong Sun. InfLLM: Training-Free Long-Context Extrapolation for LLMs with an Efficient Context Memory. In The Thirty-eighth Annual Conference on Neural Information Processing Systems, 2024.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, Song Han. DuoAttention: Efficient Long-Context LLM Inference with Retrieval and Streaming Heads. In International Conference on Learning Representations, pp. 37228–37253, 2025.

Shang Yang, Junxian Guo, Haotian Tang, Qinghao Hu, Guangxuan Xiao, Jiaming Tang, Yujun Lin, Zhijian Liu, Yao Lu, Song Han. LServe: Efficient Long-sequence LLM Serving with Unified Sparse Attention. arXiv preprint, 2025.

Feiyu Yao, Zhixiong Niu, Xiaqing Li, Yongqiang Xiong, Juan Fang, Qian Wang. An Efficient Hybrid Sparse Attention with CPU-GPU Parallelism for Long-Context Inference. arXiv preprint, 2026.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, Wangding Zeng. Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 23078–23097, 2025.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, Amr Ahmed. Big bird: transformers for longer sequences. In Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, Tuo Zhao. AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. arXiv preprint, 2023.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, Beidi Chen. H2O: heavy-hitter oracle for efficient generative inference of large language models. In Proceedings of the 37th International Conference on Neural Information Processing Systems, 2023.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, Jianfei Chen. SpargeAttention: Accurate and Training-free Sparse Attention Accelerating Any Model Inference. arXiv preprint, 2025.

AI USE STATEMENT

We used generative AI tools to translate the text (Claude Opus 5.0), to check grammar and spelling and polish the wording (GPT-5.6), to guide the drawing of the figures (Gemini), and to check the content (Grok). We did not use generative AI tools to modify code. We have reviewed all AI-assisted work: we checked every reference for hallucinated citations. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The details needed to reproduce the results of this paper, including the workload construction, the baseline rules, the metric definitions, the operating point, the evaluation protocol and the software setup, are disclosed in this appendix.

APPENDIX

Table 7 lists the contents of each part of the appendix.

Table 7: What each part of the appendix holds.

Section	Contents
Appendix A.0	Limitations
Appendix A.1	Extended related work
Appendix A.2	Motivation diagnostics per workload
Appendix A.3	Design map
Appendix A.4	Algorithm
Appendix A.5	Workload construction
Appendix A.6	Baseline rules
Appendix A.7	Metric definitions
Appendix A.8	Operating point
Appendix A.9	Software and reproduction
Appendix A.10	Evaluation protocol
Appendix A.11	Main benchmark details
Appendix A.12	Window at the budget bound
Appendix A.13	Held-out workloads
Appendix A.14	Component ablations
Appendix A.15	Budget allocation
Appendix A.16	Admission threshold
Appendix A.17	Selection persistence
Appendix A.18	Certificate soundness
Appendix A.19	Bound tightness
Appendix A.20	Per-step cost and complexity
Appendix A.21	Next steps

APPENDIX A.0: LIMITATIONS

The certificate as a training signal. The rule reads a fixed model and changes nothing about the keys it summarizes. A reader who trains long-context models would want to know whether the certificate, used as a training signal, leads a model to keys whose block boxes are tighter; training with the certificate in the loop is not part of this study.

Contexts split across devices. Every context in this study is held in one place, so each head assembles its ledger from blocks it can read directly. A reader who serves contexts too long for one device needs the blocks of a head spread over several devices and their ledgers combined; that distributed setting is not part of this study.

Verdict across tolerances. The verdict of the rule is read at the single tolerance of the operating point. A practitioner who sets a tolerance needs to know where the rule begins to refuse its selection, which a sweep over tolerances shows; this sweep can be reported on request.

APPENDIX A.1: EXTENDED RELATED WORK

Block-sparse and content-routed attention. The oldest of these lines cuts a long context into blocks and ranks the blocks by a summary of their own contents. Content-based routing clusters keys and queries and attends inside the learned clusters: the partition is a function of the content rather than of position (Roy et al., 2020). Block-max pruning keeps the blocks whose largest logit is highest, which makes the ranking quantity the cheapest upper envelope of the block's logits (Ohayon et al., 2025). Adaptive block sizing removes the fixed block length and lets the partition itself move with the content: a dense region and a sparse one are not forced into the same granularity (Liu et al., 2026). Query-aware page selection tracks the minimal and maximal key of each cache page, estimates from them how critical a page is to the current query, and loads only the top pages (Tang et al., 2024): the per-page key range it keeps is the same box a block summary is built from, spent on ranking the pages. A training-free context memory splits the distant context into blocks, represents each block by a few of its tokens and looks up the blocks relevant to the current query (Xiao et al., 2024b). A trained selector replaces the hand choice altogether and learns which tokens a query should attend over (Wu et al., 2021). All of these rank a block on a quantity whose scale carries no information about the attention mass the unselected blocks hold: two selections with the same top score can omit very different amounts of probability. Fixed patterns sit at the other end of this line: a sliding window with a few global tokens attends by position (Beltagy et al., 2020), and random connections added to it keep the universal-approximation property of full attention (Zaheer et al., 2020).

Learned and hybrid sparse patterns. A second line keeps the block structure but changes how the pattern is produced and stabilized during training. A gate placed on the sparse branch controls how much of the read-out it may carry, which keeps the optimization of a sparse attention stack from collapsing onto a single branch (Shen & Shen, 2026). Long-context stacks combine a block-sparse branch with a linear one, each covering a different range of the context, and the combination is trained end to end (Wang et al., 2026). Natively trainable sparse attention combines coarse-grained token compression with fine-grained token selection and trains the combination end to end on hardware-aligned kernels (Yuan et al., 2025), and mixture of block attention routes each query to a few key blocks through a learned gate, carrying the mixture-of-experts principle into attention (Lu et al., 2025). What these share with the previous line is that the pattern is a learned object evaluated by the loss it reaches, and what is given up when the pattern skips a region is never a quantity the run reports.

Training-free selection and cache compression. A third line works under exactly the operating constraint of this paper: the model is fixed, a budget is committed, and the rule changes nothing about the context it reads. Training-free sparse attention accelerates a fixed model by dropping blocks at inference (Zhang et al., 2025), and a unified serving stack fuses several sparsity forms behind one scheduler so that a deployment picks a budget rather than a mechanism (Yang et al., 2025). Dynamic token pruning removes positions as the sequence advances; hence the retained set shrinks with the decode (Long et al., 2025). Cache compression estimates the attention a key will receive from queries that have not yet arrived and evicts on that estimate (Devoto et al., 2025), while a simpler rule uses a key's norm alone to decide what the cache keeps (Devoto et al., 2024). A low-rank factorization goes one step further and replaces the attention the budget removes with an approximation of it (Li et al., 2025). Heavy-hitter eviction keeps a balance of recent tokens and the small share of tokens that collect most of the attention, and casts eviction as a dynamic submodular problem (Zhang et al., 2023b). A related rule rests on the persistence of importance, the hypothesis that a token pivotal at one step stays pivotal at later ones (Liu et al., 2023); Challenge #2 measures the same persistence on the retained ordering. Observation-window compression reads the attention of a window at the end of the prompt and keeps, per head, the clustered positions that window attends to (Li et al., 2024b), and dynamic sparse pre-filling assigns each head one of three recurring sparse shapes offline and builds its sparse indices at inference (Jiang et al., 2024). Across this line the quantity that is never computed is the same one: what the discarded portion was worth. A rule

that dropped nothing and a rule that dropped a third of the probability mass are therefore compared only against each other, and a practitioner who needs a guarantee at a given budget has nothing to read.

Budget allocation under attention concentration. A fourth line studies how one shared budget is split rather than how a single selection is made. Importance-driven allocation distributes a shared parameter budget across matrices by an importance score, which is the closest precedent for spending one budget where it is needed rather than uniformly (Zhang et al., 2023a), and the low-rank adaptation setting in which such budgets are usually posed is the one of Li et al. (2024a). Carried into attention, a hybrid rule spends a per-head budget beside a mandatory local window and gives the rest to the blocks its own score ranks highest (Yao et al., 2026). The quantity allocated in the first two is a rank and in the third a keep count, and in all three the axis the budget is spent against is a proxy for importance rather than an incurred cost. Key-value cache budgets are split unevenly as well: across layers, with the cache shrinking from the lower layers to the higher ones (Cai et al., 2025); across heads, by adapting each head's eviction budget (Feng et al., 2025), by profiling each head and assigning it a compression policy (Ge et al., 2024), or by separating retrieval heads that keep the full cache from streaming heads that keep only the first tokens and a recent window (Xiao et al., 2025). What makes a uniform per-head count the wrong unit is where the attention mass sits: the first positions of a long context absorb a large share of it, and that concentration is now understood as a structural property of the trained model rather than an artifact of a particular stack (Qiu et al., 2026; Son et al., 2024). Keeping the key-value states of the first tokens beside a recent window is enough for a model to stream over inputs far longer than its training window (Xiao et al., 2024a). A rule that ignores it spends its first blocks re-discovering the same positions at every head.

APPENDIX A.2: MOTIVATION DIAGNOSTICS PER WORKLOAD

Every reading in this section is taken under the plain top-k rule alone, with no admission test, no certificate, no marginal allocation and no mandatory window. The five motivation workloads read here share no key, no value and no query position with the workloads the main experiments read; the three challenges are thus read off the problem rather than off the construction. Each workload carries 8 heads over 4 layers, 64 blocks of 64 keys and 1536 query positions. The five fall into two concentration regimes that every panel of this section separates: Retr-A2 and Retr-B2 concentrate the attention mass over few blocks, while Mixed-A2, Mixed-B2 and Mixed-C2 spread it, and the workloads inside one regime read within 0.016 of each other on both readings of Table 9. Table 8 collects the three diagnostics averaged over the five workloads at the shared keep count of 8 blocks.

Table 8: The three challenges measured on the problem itself.

Challenge	Diagnostic	Mean	Low	High
#1	retained dense mass share at the shared keep count	0.684	0.554	0.871
#1	spread of true retained share within a matched observable reading	0.135	0.092	0.169
#1	heaviest single discarded block, share of dense mass	0.0147	0.0066	0.0203
#2	overlap of the early set with the later set, at the widest early window	0.209	0.117	0.273
#2	overlap of the early set with the later set, at its best early window	0.259	0.184	0.332
#2	overlap expected by chance	0.125		
#3	per-head retained share, lowest head on the workload	0.657	0.528	0.856
#3	per-head retained share, highest head on the workload	0.714	0.588	0.889
#3	per-head retained share, spread in points	5.684	3.127	7.207

Panel (a): the discarded mass is unobservable. Figure 4 is the left panel of Figure 2 at full resolution. Queries are binned into ten deciles of the quantity the rule itself can read, its unnormalized retained mass; the vertical axis is how far the true retained share still spreads inside one such bin, averaged over the bins. The spread falls as the keep count grows and does not reach zero anywhere on the sweep; hence at every budget there are queries the rule cannot distinguish that gave up different amounts of mass. Figure 5 reads the same sweep on the absolute scale, with the mean retained share and the range observed across queries at each keep count: at a keep count of 8 the range runs from 0.286 to 0.903 on the most diffuse workload while the rule's own reading stays at the dashed

line. Table 9 gives the per-workload means of both panels, together with the share of dense mass the heaviest single discarded block holds.

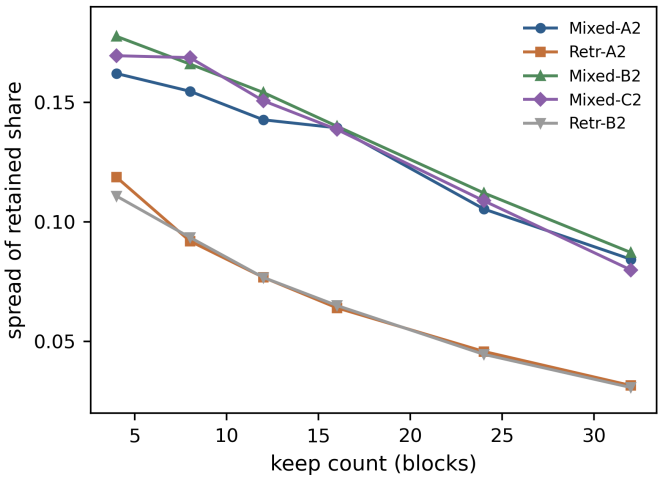

Figure 4: Spread of the true retained share inside one decile of the quantity the rule can observe, against the keep count, one line per workload.

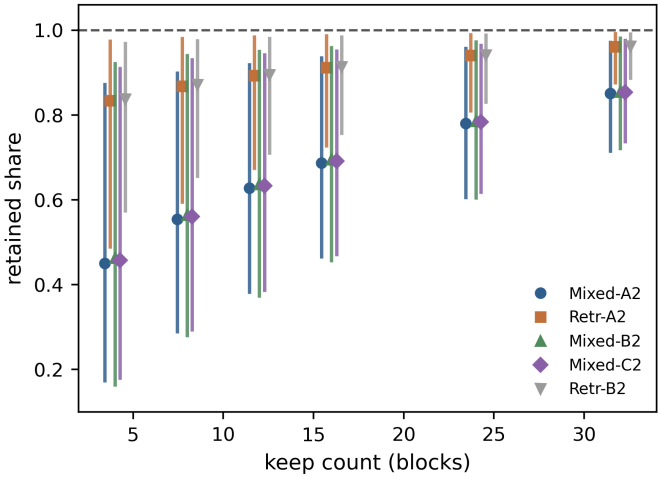

Figure 5: Mean retained share of dense softmax mass with the range observed across queries at each keep count; the dashed line is the reading the rule makes of its own retained mass.

Panel (b): the early ordering does not persist. Figure 6 is the middle panel at full resolution. The horizontal axis is the width of the early window in positions, swept over 22 widths from 2 to 40; the vertical axis is the mean fraction of the early top-k set that reappears in the later one, taken over the 32 head-and-layer positions of that workload at a keep count of 8. The dashed line is the chance overlap 0.125. No restricted-score rule is run here; hence the panel measures the ordering itself and not a rule reinforcing its own choices. The curves are not monotone in the window width, and the two windows being compared change size together with the width: the panel orders the overlap against the width without attributing the movement to either. Retr-A2 and Retr-B2 sit close to chance across the whole sweep and one of them does not clear it at every width. Table 10 gives the overlap at every width for every workload, together with the maximum over widths and the reading at the widest width.

Table 9: Challenge #1 across the keep-count sweep, one column per keep count.

Workload	keep count					
	4	8	12	16	24	32
retained share of dense mass, mean over queries						
Mixed-A2	0.451	0.554	0.628	0.687	0.780	0.851
Retr-A2	0.834	0.869	0.893	0.912	0.941	0.961
Mixed-B2	0.463	0.564	0.637	0.695	0.786	0.855
Mixed-C2	0.458	0.561	0.633	0.692	0.784	0.854
Retr-B2	0.837	0.871	0.895	0.914	0.942	0.962
spread of true retained share within a matched observable reading						
Mixed-A2	0.162	0.155	0.143	0.139	0.105	0.084
Retr-A2	0.119	0.092	0.077	0.064	0.046	0.032
Mixed-B2	0.178	0.166	0.154	0.140	0.112	0.087
Mixed-C2	0.170	0.169	0.151	0.139	0.109	0.080
Retr-B2	0.111	0.093	0.077	0.065	0.045	0.031
heaviest single discarded block, share of dense mass						
Mixed-A2	0.0308	0.0203	0.0159	0.0133	0.0099	0.0077
Retr-A2	0.0103	0.0067	0.0051	0.0042	0.0030	0.0022
Mixed-B2	0.0304	0.0199	0.0156	0.0130	0.0097	0.0075
Mixed-C2	0.0306	0.0200	0.0157	0.0131	0.0098	0.0076
Retr-B2	0.0101	0.0066	0.0050	0.0041	0.0029	0.0021

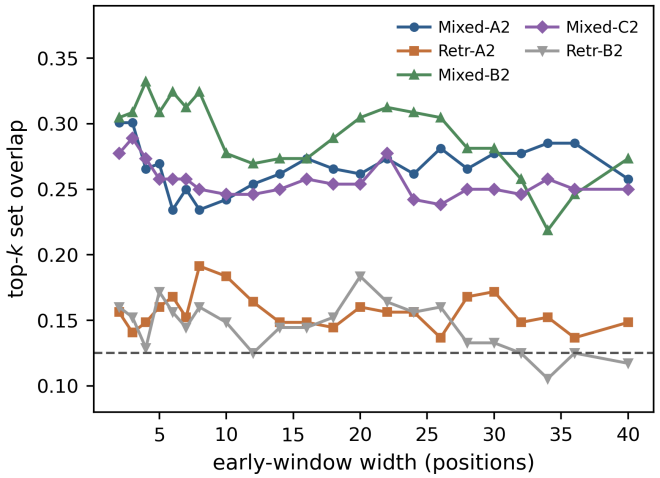

Figure 6: Overlap between the early top-k set and the later one against the early-window width, one line per workload, with the dashed line at chance.

Panel (c): one keep count buys different amounts in different heads. Figure 7 is the right panel at full resolution. Each workload contributes one curve over its 32 head-and-layer positions, ranked by the share of dense softmax mass that position retains at the shared keep count, with the workload's own mean subtracted so that the level is removed and only the spread across heads remains. All five curves lie within about -0.035 to $+0.040$ and rise smoothly rather than through a few outlying heads; the spread is thus a property of the whole head population; the concentrated workloads carry a flatter middle than the diffuse ones, a difference the absolute scale hides. Figure 8 gives the same ranking without the subtraction, where two workloads sit near 0.87 and three near 0.56. The ranking is within a workload: the spread reported in the body is the distance between the endpoints of each curve, averaged over the five workloads. Grouping the same positions by layer leaves the four layer means of a workload within 0.01 of each other while the spread across all positions of that workload is 3 to 7 points: the spread lies between heads and not between layers. Table 11 gives the lowest head, the highest head and the spread for each workload over the same 32 positions.

Table 10: Challenge #2 across the early-window sweep, one row per early-window width.

Window (positions)		Overlap with the later set				
Early	Later	Mixed-A2	Retr-A2	Mixed-B2	Mixed-C2	Retr-B2
2	46	0.301	0.156	0.305	0.277	0.160
3	45	0.301	0.141	0.309	0.289	0.152
4	44	0.266	0.148	0.332	0.273	0.129
5	43	0.270	0.160	0.309	0.258	0.172
6	42	0.234	0.168	0.324	0.258	0.156
7	41	0.250	0.152	0.312	0.258	0.145
8	40	0.234	0.191	0.324	0.250	0.160
10	38	0.242	0.184	0.277	0.246	0.148
12	36	0.254	0.164	0.270	0.246	0.125
14	34	0.262	0.148	0.273	0.250	0.145
16	32	0.273	0.148	0.273	0.258	0.145
18	30	0.266	0.145	0.289	0.254	0.152
20	28	0.262	0.160	0.305	0.254	0.184
22	26	0.273	0.156	0.312	0.277	0.164
24	24	0.262	0.156	0.309	0.242	0.156
26	22	0.281	0.137	0.305	0.238	0.160
28	20	0.266	0.168	0.281	0.250	0.133
30	18	0.277	0.172	0.281	0.250	0.133
32	16	0.277	0.148	0.258	0.246	0.125
34	14	0.285	0.152	0.219	0.258	0.105
36	12	0.285	0.137	0.246	0.250	0.125
40	8	0.258	0.148	0.273	0.250	0.117
Max over widths		0.301	0.191	0.332	0.289	0.184
At the widest width		0.258	0.148	0.273	0.250	0.117
Clears chance at every width		yes	yes	yes	yes	no

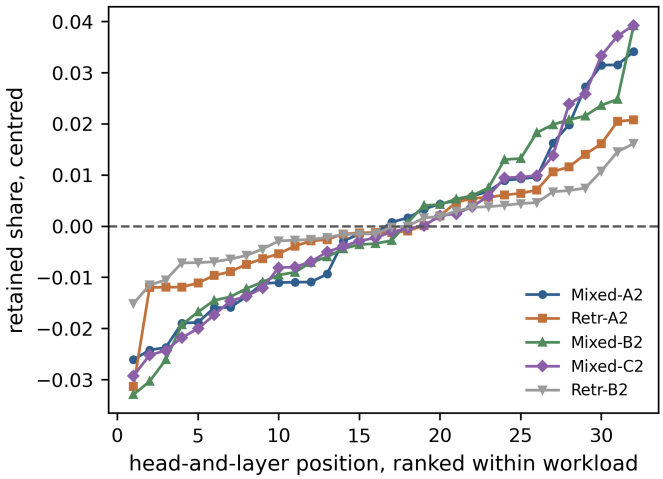

Figure 7: Retained share at the shared keep count with each workload's own mean subtracted, one curve per workload over its 32 head-and-layer positions.

APPENDIX A.3: DESIGN MAP

Table 12 pairs each component with the constraint of Eq. (1) it instantiates and the challenge it is aimed at, and names the equations that define it. The pairing is one to one in both directions: no component carries two constraints and no constraint is met by two components; hence an arm that replaces one component leaves the other two intact and the ablations read one replacement at a time.

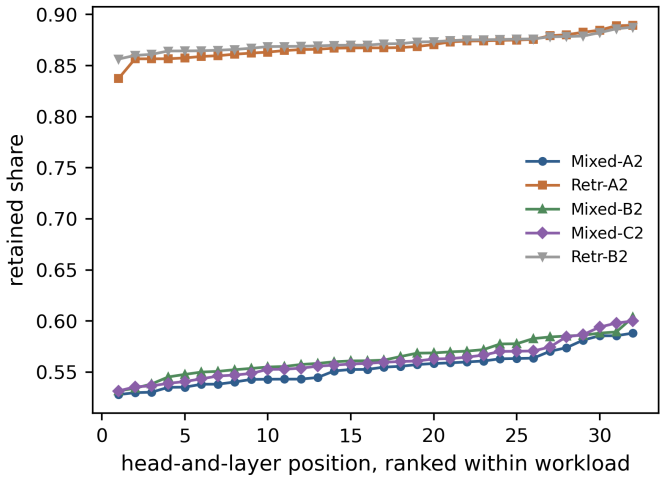

Figure 8: The readings of Figure 7 on the absolute scale, without the workload mean subtracted.

Table 11: Challenge #3, one row per workload over the same 32 head-and-layer positions.

Workload	Positions read	Lowest head	Highest head	Spread (points)
Mixed-A2	32	0.528	0.588	6.020
Retr-A2	32	0.837	0.889	5.214
Mixed-B2	32	0.532	0.604	7.207
Mixed-C2	32	0.531	0.600	6.853
Retr-B2	32	0.856	0.888	3.127
Mean over workloads	32	0.657	0.714	5.684

Table 12: Component-to-challenge design map: one formulation constraint of Eq. (1) and one challenge per component.

Component (defining equations)	Challenge addressed	Formulation constraint
Certified Block Summaries (Eq. 2 to 7)	discarded mass unobservable	certifiable $\mathcal{D} \leq \epsilon$
Certificate-Gated Selection (Eq. 8 to 10)	retained ordering unstable	admissible rule $\sigma \in \mathcal{A}$
Global Budget Allocation (Eq. 11 to 13)	per-head counts incomparable	budget $\mathcal{C}(\sigma) \leq B$

APPENDIX A.4: ALGORITHM

Algorithm 1 gives the construction in full. It runs in two phases against one budget. Calibration is performed once: the window length is read from the budget, and for every head the certificate of Eq. (5) is traced over the keep counts to give the marginal-gain curve of Eq. (12), which a greedy sweep spends the budget against, fixing the keep counts the online phase may not change. The online phase runs at each query: the box summaries are rebuilt from the block's current keys, one admission pass keeps every block whose certified mass clears a fraction of the window's retained mass, the admitted set is capped at the head's keep count, and the certificate of the resulting read-out is written into the head's ledger. Admission is independent of the keep count; hence the keep-count scan re-runs only the cap, the read-out and the certificate. On close the ledger is read against the tolerance, which is the only place the tolerance enters: above it the construction returns a refusal and no selected sets.

Algorithm 1 CertBlock: calibration once, then one pass per query

Require: key blocks $\{b\}$ with keys, values and sizes n_b; N blocks over a context of T keys; reference positions $\mathcal{Q}_{h,\ell}$; budget B over H heads and L layers; requested window w_{req}; threshold τ; tolerance ϵ
Ensure: keep counts $\{k_{h,\ell}\}$ and, at each query, the selected set $S(q)$ and its read-out $\hat{o}(S(q))$; or a refusal and no selected sets

$w \leftarrow \min\{w_{\mathrm{req}}, \lfloor (B - HL)/HL \rfloor\}$, or return a refusal when $B < 2HL$ ▷ Eq. (11)
Stage 1: calibration, run once
for each head and layer (h, ℓ) **do**
  for each $q \in \mathcal{Q}_{h,\ell}$ **do**
    $A^\star(q) \leftarrow \textsc{Admit}(q)$ ▷ independent of the keep count
    $d_k(q) \leftarrow$ the certificate from $\textsc{Certify}(q, A^\star(q), k)$, for each k from w to N
  end for
  $\hat{d}_{h,\ell}(k) \leftarrow \max_q d_k(q)$, $g_{h,\ell}(k) \leftarrow \hat{d}_{h,\ell}(k) - \hat{d}_{h,\ell}(k{+}1)$ for $k < N$, and $k_{h,\ell} \leftarrow w$ ▷ Eq. (12)
end for
while $\sum_{h,\ell} k_{h,\ell} < B$ and some head has $k_{h,\ell} < N$ **do**
  add one block to the head of largest $g_{h,\ell}(k_{h,\ell})$ among those with $k_{h,\ell} < N$ ▷ Eq. (13)
end while
Stage 2: online, at each query q of head (h, ℓ)
$A^\star(q) \leftarrow \textsc{Admit}(q)$, then $(S(q), \hat{o}(S(q)), d(q, S(q))) \leftarrow \textsc{Certify}(q, A^\star(q), k_{h,\ell})$
$D_{h,\ell} \leftarrow \max\{D_{h,\ell},\ d(q, S(q))\}$, from $D_{h,\ell} \leftarrow 0$ at the first query ▷ Eq. (7)
On close
$\mathcal{D}(\sigma) \leftarrow \frac{1}{HL} \sum_{h,\ell} D_{h,\ell}$ ▷ Eq. (7)
return $\{k_{h,\ell}\}$ with the selections if $\mathcal{D}(\sigma) \le \epsilon$, otherwise a refusal and no selected sets ▷ Eq. (1)

procedure $\textsc{Admit}(q)$
  rebuild $k_b^{\min}, k_b^{\max}, V_b$ from the block's current keys and values, then $u_b(q)$ and $m(q) \leftarrow \max_b u_b(q)$ ▷ Eq. (3)–(5); $O(Td_{\mathrm{h}})$
  $S_{\mathrm{loc}}(q) \leftarrow$ the mandatory window of length w, and $M_{\mathrm{in}}^{\mathrm{loc}} \leftarrow M_{\mathrm{in}}(q, S_{\mathrm{loc}}(q))$ ▷ Eq. (4)
  return $S_{\mathrm{loc}}(q) \cup \{\, b : n_b e^{u_b(q) - m(q)} \ge \tau M_{\mathrm{in}}^{\mathrm{loc}} \,\}$, re-testing admitted non-window blocks ▷ Eq. (8), (9)
end procedure
procedure $\textsc{Certify}(q, A^\star, k)$
  $S \leftarrow S_{\mathrm{loc}}$ with the top $k - |S_{\mathrm{loc}}|$ blocks of $A^\star \setminus S_{\mathrm{loc}}$ by certified mass ▷ Eq. (10); $O(N \log N)$
  $\hat{o} \leftarrow$ the read-out over S, with its masses $M_{\mathrm{in}}, M_{\mathrm{out}}$ ▷ Eq. (2), (4); $O(k\, n_b d_{\mathrm{h}})$
  return $\big(S,\ \hat{o},\ \frac{M_{\mathrm{out}}}{M_{\mathrm{in}} + M_{\mathrm{out}}}(\|\hat{o}\| + V_{\max})\big)$, with $V_{\max} = \max(\{0\} \cup \{V_b : b \notin S\})$ ▷ Eq. (5)
end procedure

APPENDIX A.5: WORKLOAD CONSTRUCTION

Two query constructions are used, and the name of a workload says which one it carries: a retrieval query aligns with a single block, while a mixed-document query is a Dirichlet mixture over several block directions, which is the harder of the two to select for. A name ending in 2 marks the counterpart of the workload of the same construction, built under a different seed.

A workload is a key, value and query tensor built at a fixed block structure: 8 heads over 4 layers, 64 blocks of 64 keys at head dimension 64, and 48 query positions per head, giving 32 head-and-layer positions and 1536 query positions in all. The attention mass over blocks is concentrated rather than flat, which is the regime in which a block budget is worth spending. A read-out decision is the cosine-nearest of 12 value centroids, scored against the decision the dense read-out makes on the same query, and two displacements are called indistinguishable inside a margin of 0.001. The held-out block repeats the reading at two geometries, h8/d64 and h4/d32.

APPENDIX A.6: BASELINE RULES

BSFA ranks a block by the largest logit it can produce and keeps a uniform count at every head. AB-Sparse rescores only the blocks it has materialized, refreshing them every second step; a block it has not kept thus carries no current score; it has no row in Table 18. Hybrid keeps a mandatory window and spends the rest of its count by its own score. None of the three computes the displacement bound itself: the Certificate column of Table 1 is read from the sets they commit, on the same box summaries, and their own scoring is not charged in Table 21.

APPENDIX A.7: METRIC DEFINITIONS

Write $\mathcal{Q}$ for the query positions a reading covers, $o(q)$ for the dense read-out and $\hat{o}(S(q))$ for the truncated one of Eq. (2). Dense agreement counts the positions at which the two read-outs choose the same value centroid $c(\cdot)$, and the sound rate counts the positions at which the certificate covers the displacement it is meant to bound,

$$\begin{aligned} \mathrm{Agr} &= \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbf{1}\big[\, c(\hat{o}(S(q))) = c(o(q)) \,\big], \\ \mathrm{Sound} &= \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbf{1}\big[\, d(q, S(q)) \geq \|\hat{o}(S(q)) - o(q)\| \,\big]. \end{aligned} \tag{14}$$

The displacement column is the mean of $\|\hat{o}(S(q)) - o(q)\|$ over $\mathcal{Q}$ and the certificate column the mean of the bound $d(q, S(q))$ of Eq. (5) over the same positions; their difference, averaged the same way, is the slack. Realized retention is the mean share of a head's blocks that the selected set holds, and the ledger $\mathcal{D}$ of Eq. (7) takes each head's worst position before averaging over heads, which is the quantity the tolerance is read against. Lock-in is the Jaccard overlap $|S_{\mathrm{first}} \cap S_{\mathrm{last}}|/|S_{\mathrm{first}} \cup S_{\mathrm{last}}|$ between the set S_{first} selected at the first checkpoint and the set S_{last} selected at the last. These metrics are properties of a selected set rather than of the rule that produced it.

APPENDIX A.8: OPERATING POINT

Table 13: The parameters that set the operating point.

Setting	Value
main-table workloads	Retr-A, Mixed-A
held-out workloads	Mixed-B, Mixed-C, Retr-B
budget levels	0.05, 0.10, 0.25
compared systems	AB-Sparse, BSFA, CertBlock, Hybrid
committed retention held equal across systems	0.375
admission threshold τ	0.08
read-out deviation tolerance ϵ	8.0
requested window length in blocks	8
materialized-only rescoring refresh	2
blocks per workload	64
keys per block	64
head dimension	64
heads	8
layers	4
query positions per head	48
value centroids a read-out decision chooses among	12
read-out decision rule	cosine-nearest value centroid, scored against the dense read-out's own decision
indistinguishability margin	0.001
held-out geometries	h8/d64 and h4/d32
motivation shared keep count	8
motivation keep-count sweep	4, 8, 12, 16, 24, 32
motivation early-position windows	2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 40
motivation observable-matching bins	10

Table 13 lists every quantity the run fixes before a reading is taken: the workloads each block reads, the budget levels, the rules compared and the retention held equal across them, the admission threshold and the tolerance, the workload geometry, and the decision rule a read-out is scored by. Each value is set once and held at every reading; hence a difference between two rows of Table 1 is a difference of rule rather than of operating point.

APPENDIX A.9: SOFTWARE AND REPRODUCTION

Every table, every in-prose number and every data-bearing figure is produced by one entry point, `code/run.py`, run from the package with Python 3.10 or later at the NumPy and Matplotlib versions the package pins. The only random draws are the workload constructions of `code/carrier.py`, each seeded per workload. The cost columns are counted analytically from the operation counts rather than timed; hence no wall-clock reading is reported.

The entry point writes one JavaScript Object Notation (JSON) file per result block, the table sources and the macro file the prose reads, and the figures, then copies the tables and figures into the paths the manuscript inputs; a second run thus reproduces the first.

APPENDIX A.10: EVALUATION PROTOCOL

Every rule is read at the same committed block count, such that two rules differ only in where the budget went. Calibration runs once per workload over all 1536 query positions, after which the keep counts are fixed and the online phase runs one pass per query over the same positions. The soundness check of Eq. (14) is re-run at every query of the main table rather than sampled. Each ablation holds the budget fixed and replaces exactly one quantity; hence the column it moves is the one the component was built to move.

APPENDIX A.11: MAIN BENCHMARK DETAILS

The Cert. column is the bound of Eq. (5) evaluated on each rule's own selected sets, averaged over query positions. It is reported for all four rules, because the bound is a property of the box summaries and of a selected set, not of the rule that chose the set: any block-sparse selection can be handed to it. What separates the certified rule is that it ranks blocks on their certified mass and is refused when the ledger it assembles exceeds the tolerance. At the same committed block count the certified read-out reads 1.455 and 3.578, against 1.268 and 3.198 for BSFA. The realized displacement stayed under the bound at every position, a sound rate of 1.000 and 1.000.

Table 14 carries the three columns the main table does not print. The certified mass is the share of the softmax mass each rule's selection omits, bounded rather than measured; the norm ratio reads the truncated read-out against the dense one, a value above one marking a read-out that grew; and the keep range is the spread of per-head counts the rule commits, which is a single number for BSFA and Hybrid.

Table 14: Three columns omitted from Table 1: certified mass, read-out norm ratio and per-head keep range.

Workload	Rule	Cert. mass	Read-out ratio	Keeps
Retr-A	BSFA	0.136	1.054	24–24
	AB-Sparse	0.686	0.843	21–27
	Hybrid	0.167	1.070	24–24
	CertBlock	**0.160**	**1.068**	**15–36**
Mixed-A	BSFA	0.362	1.128	24–24
	AB-Sparse	0.698	1.143	23–25
	Hybrid	0.422	1.162	24–24
	CertBlock	**0.412**	**1.179**	**11–35**

The certified rank reads 0.068 against 0.056 for BSFA, the strongest of the three baselines, and 0.137 against 0.103 on the other workload: 22.49 and 32.60 percent above it. On Retr-A it places second of the four rules and on Mixed-A third, ahead of AB-Sparse by 89.37 and 59.52 percent on the two workloads. Three parts of the construction spend the budget somewhere other than on the heaviest blocks: the window of Eq. (11) is kept unconditionally, the admission test of Eq. (8) withholds blocks from the cap, and the sweep of Eq. (13) allocates against the bound rather than against the displacement the bound covers.

The remaining columns run the same way. The per-head keep counts run from 15 to 36 on Retr-A and 11 to 35 on Mixed-A, where every baseline that is not adaptive spends exactly 24; that spread is

the allocation of Eq. (13) at work. Read as distance from the dense read-out's norm ratio of one, the read-out ratio column is closest for BSFA, 1.054 and 1.128, against 1.068 and 1.179 for the certified rule. The certified discarded mass the rule carries stands at 0.160 and 0.412 of the total, against 0.136 and 0.362 for the same baseline; on that column too the baseline reads lower.

Agreement with the dense read-out separates BSFA, Hybrid and CertBlock much less. This column counts the share of positions at which the truncated read-out picks the same value centroid the dense read-out picks. At the same budget the certified rank reads 1.000 on Retr-A, which is the reading the strongest baseline also attains, and 0.917 against 0.924 on the other, a shortfall of 0.78 percent there and 0.00 on Retr-A. What the construction adds is a selection whose tolerance is checked and can be refused, and a ranking quantity that survives while a block is unselected.

APPENDIX A.12: WINDOW AT THE BUDGET BOUND

Table 15: The certified run against the same rule with the window pushed to the bound of Eq. (11), at the same budget.

Workload	Arm	Dense agr.	Displ. mean	Cert. mean	w
Retr-A	certified run, window set from the budget	1.000	0.068	1.455	8
	ablation, window pushed to the bound	1.000	0.134	2.514	23
Mixed-A	certified run, window set from the budget	0.917	0.137	3.578	8
	ablation, window pushed to the bound	0.840	0.243	5.369	23

Eq. (11) bounds the window the budget can carry; the certified-run rows above are taken at the requested length of 8 blocks. One arm holds the rule fixed and pushes the window to that bound, 23 blocks at this budget, which leaves the ranking of Eq. (10) one block per head to place. Displacement rises on both workloads, 0.134 and 0.243 against 0.068 and 0.137, while agreement reads 1.000 and 0.840 against 1.000 and 0.917; the agreement column thus moves on Mixed-A alone. A window taken at the bound is therefore nearly the whole selection, and the readings the rule produces there are the readings of a recency rule.

APPENDIX A.13: HELD-OUT WORKLOADS

Table 3 reads Mixed-B, Mixed-C and Retr-B under two attention geometries, one with 8 heads at head dimension 64 and one with 4 heads at head dimension 32, neither of which the rule was retuned on. The certificate holds on every one of the six workload-geometry cells, at a sound rate of 1.000, and the read-out norm ratio stays between 1.010 and 1.158, the same order as on the main workloads. These results show that the construction transfers.

APPENDIX A.14: COMPONENT ABLATIONS

Three ablations replace one quantity each: the certified mass the ranking reads, the requirement that the ranking quantity survive while a block is unselected, and the marginal allocation. The first two arms keep the window, the admission test and the per-head keep counts of the certified run; hence the difference each reads is the replacement and not a second change carried with it.

Replacing the certified mass with the relevance of the block's mean key changes the displacement by 0.00002 on Retr-A and 0.00018 on the other, both within the 0.001 margin, and the mean-key arm reads 1.000 and 0.916 on agreement against 1.000 and 0.917. The bound is computable on the mean-key arm's sets as on any other: the certified mass supplies the ranking, not the bound's availability.

Replacing the certified score with a router that ranks on a score carried forward from the blocks it has already read moves the never-selected fraction by 0.2563 and 0.3115 and the displacement by 0.5009 and 0.1797. The router is seeded at its first checkpoint by the certified mass, since it has no history there, and updates only on blocks it retained; both readings run against the router. Replacing the marginal allocation with a uniform keep count is the allocation comparison already reported.

Table 16: Marginal-certified against uniform per-head allocation at three budgets; both arms commit the same block count.

			Dense agreement		Certified bound	
Workload	Budget	Blocks	Marginal	Uniform	Marginal	Uniform
Retr-A	0.05	102	1.000	1.000	6.378	6.408
	0.10	205	1.000	1.000	5.986	6.024
	0.25	512	1.000	1.000	4.124	4.212
Mixed-A	0.05	102	0.551	0.811	9.045	9.220
	0.10	205	0.684	0.819	8.582	8.696
	0.25	512	0.880	0.884	6.354	6.437

APPENDIX A.15: BUDGET ALLOCATION

Marginal-certified allocation attains a lower aggregated certified bound than a uniform per-head keep count on both workloads. At the operating point of Table 1, where the uniform keep count is run as a lesion of this component, the two bounds read 2.775 against 2.875 on Retr-A and 4.852 against 4.921 on Mixed-A; the bound columns of Table 16 carry the same ordering at each of the three budgets. It attains a lower agreement with the dense read-out in 3 of the 6 budget cells, the widest being the tightest cell of Mixed-A at 0.551 against 0.811. Both arms commit the same number of blocks at each budget; the bound columns are thus read at equal committed cost.

APPENDIX A.16: ADMISSION THRESHOLD

Table 17: The admission threshold swept over seven values at a fixed keep count.

Workload	τ	Admitted	Retention	Displ. mean	Certificate mean
Retr-A	0.000	1.000	0.3750	0.06467	1.4457
	0.005	0.998	0.3750	0.06467	1.4457
	0.020	0.957	0.3750	0.06467	1.4457
	0.050	0.824	0.3747	0.06471	1.4465
	0.080	0.702	0.3722	0.06509	1.4563
	0.150	0.505	0.3565	0.06860	1.5311
	0.320	0.294	0.2777	0.09548	2.0367
Mixed-A	0.000	1.000	0.3750	0.11942	3.5494
	0.005	1.000	0.3750	0.11942	3.5494
	0.020	0.978	0.3750	0.11942	3.5494
	0.050	0.855	0.3749	0.11945	3.5501
	0.080	0.713	0.3731	0.12019	3.5662
	0.150	0.471	0.3534	0.13110	3.7689
	0.320	0.246	0.2416	0.22134	5.1299

The admission test of Eq. (8) is swept over seven values of τ from 0 to 0.32 with the block length fixed, and, so that τ is the only quantity varying, with the keep count held at 24 blocks for every head rather than allocated. The displacements of Table 17 are therefore not the main table's, which allocates the keeps.

Up to $\tau = 0.020$ the test admits almost every block on both workloads, its admitted share falling no lower than 0.957, and the reading does not move: the realized displacement is 0.06467 on Retr-A and 0.11942 on Mixed-A at $\tau = 0$ and unchanged at every threshold up to that one. At the operating point $\tau = 0.08$ the admitted share of blocks falls from 1.000 to 0.702 on Retr-A and from 1.000 to 0.713 on Mixed-A, the realized retention falls to 0.3722 and 0.3731 from 0.375, and the displacement reads 0.06509 and 0.12019. Admitting fewer blocks raises the displacement by 0.00042 and 0.00078 at this threshold and by 0.03080 and 0.10192 across the whole sweep. At $\tau = 0.32$ the retention has fallen to 0.2777 and 0.2416; that row is outside the operating point. The one-pass property of Eq. (9) is re-checked at every query of Table 1 and held at all of them.

Table 18: Blocks a rule revisits across checkpoints, with the first-to-last Jaccard overlap.

Workload	Rule	Never selected	First-to-last Jaccard	Retention
Retr-A	BSFA	0.000	0.301	0.375
	Hybrid	0.001	0.473	0.375
	CertBlock	0.003	0.430	0.371
Mixed-A	BSFA	0.000	0.302	0.375
	Hybrid	0.001	0.474	0.375
	CertBlock	0.009	0.428	0.370

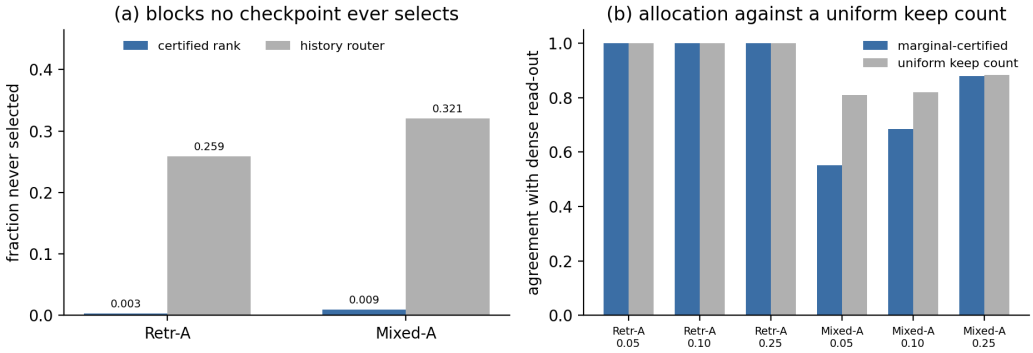

Figure 9: Two ablations at a fixed budget: (a) certified rank against a history router, on blocks never selected; (b) marginal against uniform allocation, on dense agreement.

APPENDIX A.17: SELECTION PERSISTENCE

Figure 9 shows the router arm (a) and the uniform-allocation arm (b), and Table 18 gives the per-rule readings. The certified rule leaves 0.003 and 0.009 of blocks never selected across checkpoints, against 0.000 for BSFA and 0.001 for Hybrid, and its first and last checkpoint sets have a Jaccard index of 0.430 and 0.428, against 0.301 and 0.302 for BSFA and 0.473 and 0.474 for Hybrid. The never-selected fractions are within a percentage point of each other and of zero. On the Jaccard column the certified rule sits between the two baselines: its first and last sets share more than BSFA's do and less than Hybrid's, on both workloads.

APPENDIX A.18: CERTIFICATE SOUNDNESS

Table 19: Certified against true discarded mass at five retention levels.

Workload	Retention	True discarded	Certified discarded	Bound/true
Retr-A	0.094	5.1	12.6	2.477
	0.250	3.1	7.6	2.471
	0.375	2.1	5.1	2.467
	0.500	1.3	3.3	2.464
	0.750	0.4	1.0	2.459
Mixed-A	0.094	81.5	168.1	2.063
	0.250	52.4	107.8	2.058
	0.375	36.9	75.9	2.055
	0.500	25.2	51.7	2.052
	0.750	8.7	17.8	2.046

The bound is read against two different quantities. The first is the displacement it bounds, taken from the run of Table 1 itself: across all 1536 positions per workload the count of positions at which the certificate fell below the realized displacement is 0. This is the same run and the same selected

sets as the main table. The second is the mass the bound stands on, read at selections the main table does not use: 5 retention levels, 0.094 through 0.750; that block, Table 19, reads 384 of the 1536 positions rather than all of them. The True discarded and Certified discarded columns carry the mass comparison: at every one of the ten rows the certified discarded mass stands above the mass actually discarded, and the Bound/true column is that ratio. The sound rate at those positions, the rate at which the certificate stands at or above the realized read-out displacement, reads 1.000 at every level. The derivation makes both inequalities hold for any workload; these rows thus read the implementation against the derivation: a row falls below one only if the box of Eq. (3) is built from stale keys, or if the read-out and the certificate are taken at different selected sets.

APPENDIX A.19: BOUND TIGHTNESS

Table 20: Certificate slack as within-block key spread is varied.

Key spread	Slack mean	Cert. mass frac.	Displ. mean	Sound rate
0.020	2.000	0.602	0.382	1.000
0.050	2.150	0.640	0.382	1.000
0.100	2.381	0.698	0.380	1.000
0.200	2.774	0.797	0.372	1.000
0.350	3.167	0.897	0.364	1.000
0.600	3.467	0.970	0.342	1.000

The price has two parts. One is the mass substitution: the certified discarded mass exceeds the mass actually discarded by a factor of 2.046 to 2.063 on one workload and 2.459 to 2.477 on the other. The other is the pair of norm bounds inside the bracket of Eq. (5), which replace the distance between the read-out and the discarded values by the sum of two radii. Together they put the bound at 21.4 and 26.2 times the mean displacement. Table 20 reads one quantity against the slack, which is the certificate minus the realized displacement and hence a difference rather than the ratio the Bound/true column carries: as the within-block key spread is widened from 0.020 to 0.600 the slack grows from 2.000 to 3.467, and the narrowest spread read already carries 2.000 of it.

APPENDIX A.20: PER-STEP COST AND COMPLEXITY

Table 21: Multiply-accumulates per query against a dense pass, under both charging conventions. AB-Sparse has no row.

		Multiply-accumulates per query			Fraction of dense	
Workload	Rule	Bound, admission	Read-out	Summary, amort.	Amortized	Charged once
Retr-A	BSFA	0.0	98304.0	0.0	0.375	0.375
	Hybrid	0.0	98304.0	0.0	0.375	0.375
	CertBlock	4544.0	97304.0	5461.3	0.409	1.389
Mixed-A	BSFA	0.0	98304.0	0.0	0.375	0.375
	Hybrid	0.0	98304.0	0.0	0.375	0.375
	CertBlock	4544.0	97096.0	5461.3	0.409	1.388

The cost account of Table 21 carries the two terms the construction adds, the box rebuild and the admission pass, rather than netting them away, and it is printed under both conventions. Amortizing the summary over the queries of a step, the certified read-out spends 0.409 of a dense pass where BSFA spends 0.375. Charging the summary once per query instead, the same read-out spends 1.389, which is more than dense. Which convention is right depends on how often the box is rebuilt relative to the queries it serves; both are therefore reported. Table 22 separates the terms behind those two numbers, marking each as saved, added or retained. Every term except the bound curve is counted per query: the removed attention term is linear in the discarded blocks.

Below a budget of 64 blocks the window length of Eq. (11) is undefined, and no reading is taken there.

Table 22: Per-step cost account of CertBlock at one query, relative to a dense pass.

Term	Scaling	Status
Dense attention (removed)	linear in discarded blocks $\times n_b \times d_{\mathrm{h}}$	saved
Box summary and value radius	linear in context length $\times d_{\mathrm{h}}$	added
Logit bound $u_b(q)$ over all blocks	linear in blocks $\times d_{\mathrm{h}}$	added
Admission, ranking and cap	blocks per head, times its logarithm	added
Bound curve of Eq. (12)	once per allocation, not per query	added
Read-out over $S(q)$	linear in retained blocks $\times n_b \times d_{\mathrm{h}}$	retained

APPENDIX A.21: NEXT STEPS

The first of the two next steps is a budget at which the mandatory window is a smaller share of each head's keeps. The second is a tighter box: Table 20 reads the slack as the key spread is widened from 0.020 to 0.600, where it grows from 2.000 to 3.467.