LOST IN A FEW HEADS: DIAGNOSING AND REPAIRING BLOCK SELECTION IN SPARSE ATTENTION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse attention reduces the compute and memory cost of long-context inference, but already-trained sparse attention models may lose substantial accuracy at their deployed selection budgets. We localize where this loss occurs. Using exact attention mass selection as an oracle diagnostic per token and per head, we locate the multi-key retrieval loss in the InfLLM-V2 family to three or four of its 64 key-value heads, the same heads in four checkpoints of one backbone. These blind heads attend to the needle, yet their selector misses it likely because the mean-pooled key summary scored by the selector dilutes single decisive token. A retrieval-head score ranks heads whose selector already finds the needle above the blind heads. Once the heads are found, the loss can be repaired without training most simply by using full attention in them, costing the least for a single sequence. An alternative is a scorer ranking blocks with each query head's full softmax over a 4-bit copy of the keys while keeping the shipped budget and reading an eighth of the bytes that full attention reads in blind heads. On prompts held out from every design we test, the two repairs recover 31.4 and 30.1, respectively, out of the 33.2 points on RULER MK3 multi-key retrieval that InfLLM-V2-8B loses, and 41.0 and 41.1 of 41.8 on MiniCPM4.1-8B at 64K. On four other selector designs, the scorer recovers most of the loss by acting in many or all of their heads.

1 INTRODUCTION

Large Language Models (LLMs) are increasingly applied to long context tasks, including multi-turn chat conversations, long document understanding, agentic workflows, and multi-step reasoning (Anthropic, 2026; DeepSeek-AI, 2026; Google DeepMind, 2026; Kimi Team, 2026; OpenAI, 2026; Zhipu AI, 2026). However, full attention exhibits quadratic complexity in the context length. As context windows grow from 32K to 128K tokens and beyond, attention comes to dominate both the latency of the prefill and the memory that the key-value cache holds (Dao et al., 2022; Dao, 2023; Shah et al., 2024; Sun et al., 2025b).

Recently, sparse attention has become a promising alternative to full attention, now trained into released models (Sun et al., 2025b). Token-level sparse attention mechanisms score every past key with a lightweight indexer and only select the highest-scoring keys (DeepSeek-AI, 2025; GLM-5 Team, 2026). Block-level designs instead summarize contiguous blocks once and select the top-k whole blocks, which is far cheaper to score and maps better onto the memory system (Yuan et al., 2025; Lu et al., 2025). Across both families, the attention that follows selection is exact, and the selector runs at a budget fixed by the model producers.

However, a fixed budget introduces a subtle failure in sparse attention. Because attention after selection is exact, most of the model's accuracy loss comes from poor selection, and a budget that is adequate on average can still be spent on the wrong blocks for a particular query. Sparsification can modify what a model relies on, in ways that aggregate accuracy hides (Ren et al., 2026). For instance, on RULER's multi-key retrieval at 32K, InfLLM-V2-8B's own report puts its sparse attention mode far below the same model run with dense attention (Hsieh et al., 2024; Zhao et al., 2026). We measure similar losses for MiniCPM4.1-8B (MiniCPM Team, 2025) and the SeerAttention-R gates (Gao et al., 2026), all within their native context ranges.

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.

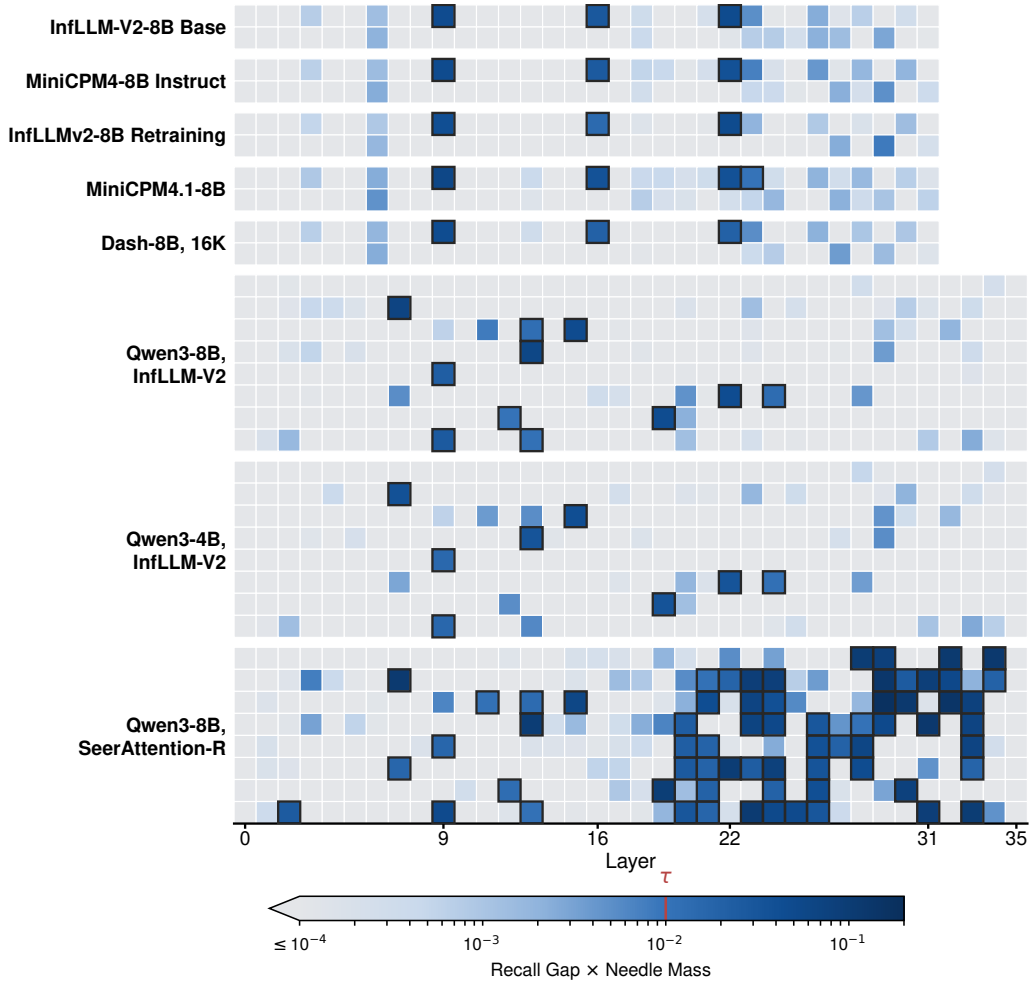


Figure 1: **The same heads are flagged in every checkpoint of a backbone.** Per key-value head, the needle recall that the shipped selector loses against exact mass, times the attention the head puts on the needle block. 50 calibration prompts at 32K (16K for Dash-8B). Bands are the five MiniCPM4-8B checkpoints, head 0 above head 1, the InfLLM-V2 selector trained on Qwen3-8B and 4B, and Qwen3-8B under the SeerAttention-R gate at a budget of 4096 tokens, heads 0 to 7 from the top. Outlined heads pass the threshold $\tau = 0.01$, three or four per MiniCPM4-8B checkpoint, 11 and 8 of 288 on the two Qwen3 models (the smaller set inside the larger) and 66 of 288 for the SeerAttention-R gate. On Dash-8B, flagged heads carry only about a quarter of the loss (Table 5). Scores at or below 10^{-4} share the lightest color.

Solutions that do not locate the loss are costly. A larger budget reads more blocks in every head at every query, and lowers the savings that sparse attention was adopted for. Retraining the model or its selector is expensive, and a selector trained to match another one learns its failures as well (Fu et al., 2026). Instead, we ask where the loss is made. Using exact attention mass as an oracle diagnostic, we localize the accuracy loss of the InfLLM-V2 family to three or four blind heads, in four checkpoints of the same backbone. These heads attend to the needle, but their selector misses it because mean-pooling keys in a block dilutes single decisive tokens. We further localize the loss to either the prefill, question prompt, or decode tokens of a given task. Once the blind heads are known, training-free fixes recover most of the loss, most simply by enabling full attention in them. We also explore replacing the shipped selector with exact attention mass on a 4-bit copy of the keys while keeping the shipped budget, scanning an eighth of the bytes that full attention in blind heads reads. Both repairs change only the flagged heads, and a one-time calibration per checkpoint suffices to find such heads. We build our repairs on InfLLM-V2-8B, extend them to the three other checkpoints that share this selector, to NOSA-8B (Huang et al., 2025), and to four other selector designs (Gao et al., 2024; 2026; Huang et al., 2026; Yuan et al., 2025), recovering most of the loss.

In summary, our contributions are:

- **Localizing the loss.** Exact attention mass selection at the deployed budget localizes the multi-key deficit of the InfLLM-V2 family to three or four blind heads of its 64 key-value heads, and to the query tokens at which the loss is made. A retrieval score measures the head, not the selector, so it ranks the blind heads below heads whose selector already finds the needle (Section 5).
- **A resolution account.** An idealized two-block model extending Xiao et al. (2025) shows that a score increasing in the mean logit of a pooling window misranks needles that exact attention mass would rank correctly. On the probed tokens, two of the three flagged heads exhibit this behavior, while quantized keys keep the ranking (Section 5, Appendix D).
- **Two training-free fixes.** Full attention in the flagged heads costs the least for a single sequence while a soft vote over a 4-bit copy of their keys keeps the shipped budget, and both recover most of the loss on held-out prompts for every design we test. On four other selector designs, the scorer needs many or all heads (Sections 6 and 7).

2 Related Work

Trainable sparse attention. Some recent works train the selector with the model. Selection is either over blocks in NSA (Yuan et al., 2025), MoBA (Lu et al., 2025), InfLLM-V2 (Zhao et al., 2026; MiniCPM Team, 2025), DashAttention (Huang et al., 2026), NOSA (Huang et al., 2025) and MiniMax Sparse Attention (Lai et al., 2026), or over single tokens in DSA (DeepSeek-AI, 2025) and LongCat (Zan et al., 2026). Others train only a selector for a frozen model, such as a block gate in SeerAttention (Gao et al., 2024; 2026), a distilled block router in SpotAttention (Ahmad & Yun, 2026), and a forecast of the next layer's blocks in SparDA (Fu et al., 2026).

What a block summary loses. A block summary can hide the block's heaviest token (Xu et al., 2026; Hwang, 2026; Kalyanarangan, 2026), which RBS-Attention (Song et al., 2026) calls mean dilution and rescues it at prefill. Xiao et al. (2025) show a centroid router's signal-to-noise ratio falls with the block size. COBS (Tian et al., 2026) shows that selecting the blocks of largest attention mass lets block sparse attention match full attention, and that mean-key selectors approximate it to first order. Wang et al. (2026a) use the same above oracle, averaged over heads, to separate the budget from the indexer's error. Quest (Tang et al., 2024b) bounds a block by the per-channel extremes of its keys. Multipole (Hooper et al., 2025), Prism (Wang et al., 2026b) and COBS all improve the block summary, and LOCKS (Hwang, 2026) ranks pages by a log-sum-exp mass from one compact code per page, and proves that a code shared by a page's keys cannot separate some pages that differ inside. Proposition 1 gives both sides: the margin at which exact mass keeps the needle block and the margin at which a mean score loses it to competitor blocks.

Low-bit scoring and retrieval heads. Scanning a compressed copy of the keys is explored for dense models (Yang et al., 2024; Singhania et al., 2024; Ji et al., 2026; Liu et al., 2026b; Kalyanarangan, 2026; Wang et al., 2025). Twilight (Lin et al., 2025) and SpeCache (Jie et al., 2025) rank a base selector's picks or an offloaded cache with a low-bit copy of the keys. HISA (Xu et al., 2026) and self-indexing attention (Yang et al., 2026) replace a deployed token indexer in every head. A few heads carry retrieval in dense models (Wu et al., 2024b; Xiao et al., 2024), and DuoAttention and RazorAttention (Tang et al., 2024a) give them the full cache. RTPurbo (Zhou et al., 2026) gives full attention at prefill and a trained indexer at decode to heads found with a needle probe. AB-Sparse (Liu et al., 2026a) calibrates a block size per head, and Lin et al. (2026) argues that the retrieving heads vary within a generation. We fix the heads once per checkpoint of a sparse-trained model.

3 Preliminary

Full attention with grouped queries. For one layer, query token t, query head g and head dimension d, softmax attention computes $\boldsymbol{o}_t^{(g)} = \sum_{j \le t} a_{t,j}^{(g)} \boldsymbol{v}_j$ with $a_{t,j}^{(g)} = \mathrm{softmax}_j(\boldsymbol{q}_t^{(g)\top} \boldsymbol{k}_j / \sqrt{d})$, at a cost quadratic in the sequence length T. Grouped Query Attention (Ainslie et al., 2023) lets G query heads share one key-value head.

Block sparse attention. Sparse attention chunks the keys into contiguous blocks $\mathcal{B}_1, \mathcal{B}_2, \ldots$, each of L tokens. A block sparse selector gives each query a forced set $\mathcal{F}$ of blocks it always reads: the

initial block $\mathcal{B}_{init}$, a local window $\mathcal{B}_{local}$, and the query's own block $\mathcal{B}_{query}$. It fills K free slots with the blocks of highest score c_b, $\mathcal{S} = \mathrm{TopK}_{b \notin \mathcal{F}}(c_b, K)$. Attention is then exact over the keys of $\mathcal{F} \cup \mathcal{S}$, and the budget $|\mathcal{F}| + K$ is fixed. Block selectors differ only in how the score c_b is computed.

Pooled-key selection. InfLLM-V2 (Zhao et al., 2026) scores blocks from mean-pooling of keys over windows W_m of w consecutive keys. Each query head computes a softmax over the pooled keys, the heads of the group are summed, and a max-pool maps the window scores back onto blocks,

$$\bar{\boldsymbol{k}}_m = \frac{1}{w} \sum_{j \in W_m} \boldsymbol{k}_j, \qquad c_b^{\mathrm{pool}} = \max_{m \in \mathcal{W}(b)} \sum_{g=1}^{G} \mathrm{softmax}_m \left(\frac{\boldsymbol{q}_t^{(g)\top} \bar{\boldsymbol{k}}_m}{\sqrt{d}} \right), \tag{1}$$

where $\mathcal{W}(b)$ is the set of windows that the kernel's max-pool assigns to block b.

Exact mass selection. Deployed selectors estimate the attention mass that a block would receive (Tang et al., 2024b; Yuan et al., 2025; Gao et al., 2024). The selection criterion that keeps the most attention mass for query t at a given budget is exact mass selection,

$$c_b^{\mathrm{exact}} = M_b = \sum_{g=1}^{G} \sum_{j \in \mathcal{B}_b} a_{t,j}^{(g)}, \tag{2}$$

which we use as an oracle in Section 5.

4 Setting and Experimental Setup

We study deployed sparse attention checkpoints at their shipped attention budget. Every sparse variant of a family reads the same number of key-value blocks per query, through the model's own kernel and cache. Only the selected block indices change.

Systems. We build on InfLLM-V2 (Zhao et al., 2026), whose selector combines fixed initial blocks, a local window, and top-k blocks selected by Equation 1 (Xiao et al., 2023; Yuan et al., 2025; Lu et al., 2025). We study four checkpoints with this mechanism: the base model, the MiniCPM4-8B instruct model (MiniCPM Team, 2025), a separate InfLLMv2-8B retraining of the same backbone by Huang et al. (2026), and MiniCPM4.1-8B. Each checkpoint has a full attention setting that differs by one configuration flag on identical model weights, and Appendix A gives their architecture and selector settings. We extend the study to NOSA-8B (Huang et al., 2025), which adds a query-agnostic eviction head on top of InfLLM-V2. We also apply our method to other selector designs, including the SeerAttention-R gates on Qwen3-4B, 8B and 14B (Yang et al., 2025a), the SeerAttention gate on Llama-3.1-8B (Grattafiori et al., 2024), and Dash-8B and NSA-8B, which Huang et al. (2026) continued from MiniCPM4-8B.

Evaluation. Our main task is MK3, RULER's multi-key needle task (Hsieh et al., 2024) where the haystack is entirely distractor needles and on which selectors fail the most. We also report the other RULER tasks, HELMET (Yen et al., 2024), and five benchmarks of real data. Every comparison in our harness is paired on identical prompts from one seed of RULER's generator, which also guided most design choices. The headline settings are repeated with five seeds, and the abstract quotes the means of the four held out from those choices (Appendix B). Appendix A states which differences a paired test separates from zero. All variants act through a hook on the selector's output that reproduces the released selector's selections exactly when the hook is inactive (Appendices A, B). Unless stated otherwise, full attention in the flagged heads acts at every token.

5 Diagnosing the Deficit

The instrument. For each query token, layer, and key-value head, we use exact attention mass selection to compute the true attention mass of every candidate block, summed over the query heads that share it. Summing per-head masses is the head soft vote of Wu et al. (2024a), and the instrument is a diagnostic of selector failures, not a deployed path.

Table 1: **Where the multi-key loss is made.** Each row applies Equation 3 at the tokens and heads its label names and keeps the shipped selector elsewhere. Exact mass approaches full attention as $\mathcal{T}$ grows. Exact mass at every token of the flagged blind heads alone reaches full attention. In all tables, unless stated otherwise, **bold** is best number and underlined is second best, excluding dense.

Selection	InfLLM-V2-8B, 32K	MiniCPM4.1-8B, 32K	MiniCPM4.1-8B, 64K
Sparse	65.5	82.0	55.5
Exact mass at decode	90.0	87.5	79.0
Exact mass at decode and window	95.5	**100.0**	96.0
Exact mass at every token of flagged heads	**98.5**	**100.0**	**97.0**
Exact mass at decode tokens of flagged heads	89.5	87.5	78.5
Dense at every token of flagged heads	98.0	100.0	97.5
Dense at every token of other heads	61.0	88.0	60.0
Dense	98.5	100.0	97.0

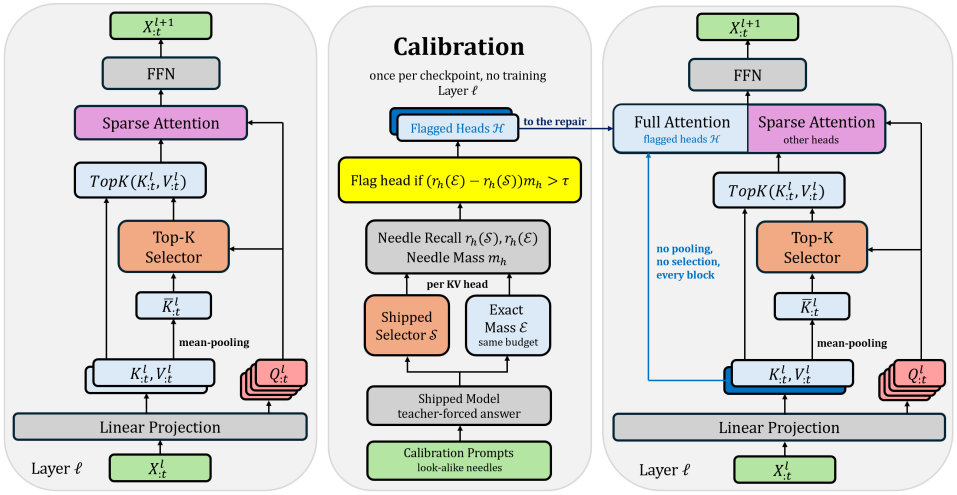
Formally, the instrument takes a set $\mathcal{T}$ of query tokens, from the prompt and from the generation, and a set $\mathcal{H}$ of key-value heads. At a token t, it gives each head the selection

$$\mathcal{S}_t = \begin{cases} \mathrm{TopK}_{b \notin \mathcal{F}}\big(M_b,\, K\big) & \text{if } t \in \mathcal{T} \text{ and the head is in } \mathcal{H}, \\ \mathcal{S}_t^{\mathrm{ship}} & \text{otherwise,} \end{cases} \tag{3}$$

where M_b is the exact mass of block b at token t from Equation 2 and $\mathcal{S}_t^{\mathrm{ship}}$ is the shipped selector's choice. The forced set $\mathcal{F}$ and the budget K stay those of the original model. Exact mass keeps the most attention mass at the shipped budget. Hence, the accuracy that Equation 3 recovers over the shipped selector estimates the loss caused by the choice of blocks, at the tokens and heads in $\mathcal{T}$ and $\mathcal{H}$. The recovered accuracy should be treated as a mass-optimal reference rather than a bound on total accuracy. In Table 1, $\mathcal{T}$ grows from the decode tokens to the last 128 prompt tokens, which hold the question, in every head and then to every token in the flagged heads alone. Dense rows in Table 1 are full attention without selector, while exact mass is used as a criterion in sparse selection.

The loss is a selection loss. Table 1 applies the instrument by token, on RULER's MK3, at the indicated context length. On these checkpoints, within their native ranges, exact mass selection at the shipped budget reaches full attention on this seed and comes within a few points of it over five seeds (Appendix B). So the budget suffices, and the loss lies in which blocks are chosen. On the retraining's smaller budget and beyond the native range, part of the loss is budget (Table 19). On InfLLM-V2-8B, exact mass at the decode tokens alone recovers most of the deficit. On MiniCPM4.1-8B, the question tokens carry a larger share of the loss, and exact mass at the decode tokens alone stays far below full attention. A repair must therefore also act inside the prefill.

The loss lives in three or four heads. To find heads that read the needle but whose selector misses it, we run a needle calibration probe with RULER MK3 prompts on each key-value head. We teacher-force the answer tokens and flag a head h when $\big(r_h(\mathcal{E}) - r_h(\mathcal{S})\big)\, m_h > \tau$ for a threshold τ. The difference $r_h(\mathcal{E}) - r_h(\mathcal{S})$ is the needle recall that the shipped selector loses against exact mass at the same budget. The score is weighted by m_h, the attention the head puts on the needle block (Figure 1, Equation 7 in Appendix C). The same set $\mathcal{H}$ enters Equation 3 in Table 1 and the repair of Section 6, and one threshold τ is used for every checkpoint we probe (Appendix C). On InfLLM-V2-8B, the calibration returns the first KV head of layers 9, 16, and 22, with flagging score far above other heads (Table 7). The same three are flagged from each of five calibration seeds, at 16K, and from held-out JSON key-value data outside RULER. Calibration at 64K flags one more head (Appendix C). The calibration returns the same three for the instruct checkpoint, for the retraining, and for Dash-8B, which has a different selector and makes most of its loss elsewhere (Section 7). The same three plus the first head of layer 23 are flagged for MiniCPM4.1-8B. At the decode tokens, exact mass in the flagged heads alone recovers as much as exact mass in every head, and full attention in the flagged heads reaches full attention. However, full attention in all the other heads, with the flagged ones left to the shipped selector, recovers no significant part of the loss on InfLLM-V2-8B and on MiniCPM4.1-8B at 64K, and only a third of it at 32K (Table 1). A scan of single heads without the probe finds the same three, and three random heads change nothing (Appendices B and C). These flagged heads attend to the needle whenever its block is selected, but their selector does not select it, so under the shipped selector they are blind to it. As such, we call them ***blind*** heads. Flagging by needle mass or by the retrieval score of Wu et al. (2024b) picks mostly heads whose selector already



(a) InfLLM-V2 (b) Calibration (c) Repair

Figure 2: **Method overview.** (a) InfLLM-V2 scores each block from mean-pooled keys, gathers the top-k blocks and attends to them in every head. (b) A needle probe, run once per checkpoint, compares the shipped selection with exact attention mass at the same budget and flags the blind heads, whose recall gap times needle mass exceeds τ. (c) In the blind heads, full attention reads every block, and every other head keeps the shipped selector. A 4-bit scorer that keeps the shipped budget can replace full attention in those heads.

finds the needle, and scores far lower (Table 13). Our scoring function is an instance of a general score that can be modified to find heads deficient in other capabilities (Appendix C).

Pooling loses the needle. In the flagged heads of layers 9 and 16 of InfLLM-V2, only a few of the group's query heads put appreciable attention on the needle block. Scored alone, each of them usually finds the needle block under its own token-level attention but misses it under its softmax over the mean-pooled keys (Table 12). Neither another aggregation over the group nor Quest's bound helps (Table 15). Let a block hold one needle key whose logit exceeds its neighbors' by Δ, and let a competitor block have the logit of every key elevated by δ. Exact mass ranks the needle block first whenever $\Delta > \delta + \log L$, while any score that increases with the mean logit over a window of w keys ranks the competitor first whenever $\Delta < w\delta$, so every needle in between is lost to pooling (Proposition 1). The window widens with the number of competitors, as the deficit does with length (Figure 5), and closes when no competitor resembles the needle. This is why the calibration haystack decides which heads are flagged. On the probed tokens, look-alike blocks raise the pooled cutoff in every head. In the flagged heads of layers 9 and 16, the rest of the needle's block stays flat (the proposition's case), so its pooled window falls below the cutoff while exact mass keeps the block. In the heads whose selector finds the needle, every key of the line containing the needle has a raised logit, so the line's best window clears the cutoff (Table 10). Quantizing the keys instead moves each block score by at most its largest logit error (Proposition 2). On the probed tokens, the needle block keeps its rank with keys quantized to a few bits and loses it with keys pooled, so a selector can spare precision but not resolution (Appendix D).

6 Training-Free Fixes for the Flagged Heads

Once the flagged heads are known, the simplest fix is giving them full attention, as DuoAttention and RazorAttention do for retrieval heads. The fix reads every key and value of those heads at every step, which costs little for a single sequence and grows with batch and context. The diagnosis also points to a repair that keeps the shipped budget and token resolution at reduced precision, changing only the top-k path in the flagged heads.

Calibration. The probe of Equation 7 runs once per checkpoint on calibration prompts at 32K, or 16K for Dash-8B, takes minutes on one GPU, and trains nothing. This is similar to the common head-level calibration on a small synthetic retrieval set (Xiao et al., 2024; Wu et al., 2024b; Jiang et al., 2024). Lower thresholds τ move the accuracy by at most a point and a higher one that drops

Table 2: **Multi-key retrieval (MK3) at the shipped budget**. First of the five evaluation seeds of Table 6. "Exact Mass" acts at the decode tokens of every head. "Dense, Flagged Heads" gives each checkpoint's flagged heads full attention at every token. "Decode", "+ Question" and "Every Token" act at the decode tokens, also at the last 128 prompt tokens, and at every token respectively, in each checkpoint's flagged heads.

						Training-free Repair		
Model	Context	Dense	Sparse	Exact Mass	Dense, Flagged Heads	Decode	+ Question	Every Token
InfLLM-V2-8B	32K	98.5	65.5	90.0	**98.0**	90.0	94.0	**98.0**
	64K	82.5	16.0	53.0	**75.5**	52.0	67.0	66.0
MiniCPM4-8B Instruct	32K	94.0	52.0	70.5	93.5	71.0	**94.0**	93.5
MiniCPM4.1-8B	32K	100.0	82.0	87.5	**100.0**	87.5	99.5	**100.0**
	64K	97.0	55.5	79.0	**97.5**	80.0	96.5	**97.5**
InfLLMv2-8B Retraining	32K	92.0	23.5	46.0	**90.5**	46.0	76.5	84.0

a head slightly costs accuracy. Probes without look-alike distractors, such as a single needle, flag other heads (Appendix C).

Side cache. Here we describe the alternative repair to full attention in blind heads. For each flagged head, we keep a 4-bit copy of its keys, used only for scoring. Each channel c, a coordinate index in the dimension of key vectors, is standardized by its mean μ_c and standard deviation σ_c, and quantized uniformly with the step Δ_b that is optimal for a Gaussian source at b bits (Max, 1960),

$$\hat{\boldsymbol{k}}_{j,c} = \boldsymbol{\mu}_c + \sigma_c \Delta_b \, z_{j,c}, \qquad z_{j,c} = \mathrm{clamp}\Big(\Big\lfloor \frac{\boldsymbol{k}_{j,c} - \boldsymbol{\mu}_c}{\sigma_c \Delta_b} \Big\rfloor + \tfrac{1}{2},\ -2^{b-1} + \tfrac{1}{2},\ 2^{b-1} - \tfrac{1}{2} \Big). \tag{4}$$

$z_{j,c}$ is the b-bit code stored for $k_{j,c}$ and $\hat{k}_{j,c}$ is the reconstructed key used in scoring blocks. Standardizing each channel lets one step fit channels whose spreads differ widely, as in per-channel key caches (Liu et al., 2024b), and Δ_b balances rounding against clipping for a unit Gaussian. Treating each standardized channel as a Gaussian source is a default that assumes nothing of the channel beyond its mean and standard deviation, and deviations from this model barely matter (Appendix E). The clamp can only change which blocks score highest, since attention reads the full-precision keys and always reads the forced blocks, the attention sink (Xiao et al., 2023) among them. At the repair's precision, the choice of grid, the step and the range the codes cover, barely matters (Appendix E). We use $b = 4$, at which the scorer keeps the needle block as well as exact mass on the probed tokens (Table 11). When the repair acts at every prompt token, μ_c and σ_c come from a fixed prefix of the prompt, so the code of a key never depends on later tokens, and otherwise from all cached keys. The copy adds under one percent to the KV cache (Appendix F).

Selection. At every token of a flagged head, in the prefill and at decode, each query head of the group takes a softmax over the low-bit keys up to its own position. The distributions are summed within each block, and the forced blocks and the best-scoring free blocks fill the shipped budget. Every other head keeps the shipped selection, and the kernel and the cache are unchanged. A cheaper repair variant acts only at the decode tokens and the last prompt tokens, SnapKV's observation window (Li et al., 2024). The repair is the instrument of Equation 3 with $\mathcal{H}$ being the flagged heads and the exact mass M_b replaced by its value on the 4-bit keys,

$$\hat{M}_b = \sum_{g=1}^{G} \sum_{j \in \mathcal{B}_b} \mathrm{softmax}_j \left(\frac{\mathbf{q}_t^{(g)\top} \hat{\mathbf{k}}_j}{\sqrt{d}} \right), \tag{5}$$

where $\hat{\mathbf{k}}_j$ is the 4-bit copy of key j. The three variants of Section 7 differ only in $\mathcal{T}$.

7 Results

Both fixes recover most of the loss. On the seeds of Tables 2 and 19, full attention in the flagged heads reaches Dense on InfLLM-V2-8B at 32K, on MiniCPM4.1-8B and on NOSA-8B, and the repair at every token matches it on the first two. Over the five seeds of Table 6, both sit a few points under Dense on InfLLM-V2-8B. On the four held-out seeds, the question-token variant recovers about the same as the every-token variant with half the added time to the first token (Appendices B and F). The same holds on the instruct checkpoint and on MiniCPM4.1-8B with four flagged heads, and the scorer in the other head of those three layers changes nothing (Table 13). On the retraining

Table 3: **Against published selectors** on the two checkpoints that Fu et al. (2026) evaluate. RULER MK3 and the mean of RULER 12 tasks at the shipped budget. Quest, Double Sparsity, Loki, and SnapKV act in every head at the decode and question tokens. "Loki, flagged heads" is at every token of the flagged heads at the repair's bytes. Bytes per cached token is what a method reads per cached token and decode step beyond the blocks every sparse method reads, for MiniCPM4.1-8B / NOSA-8B where the two differ. The whole cache is read for Dense. Settings in Appendix E.

	MiniCPM4.1-8B, 64K		NOSA-8B, 32K		Bytes per
	MK3	RULER	MK3	RULER	Cached Token
Dense	97.0	87.1	94.0	88.8	32,768
Sparse	55.5	79.5	18.5	69.3	1,024
Quest	22.0	75.0	18.5	72.7	512
Double Sparsity	34.0	73.2	29.5	72.3	2,048
Loki	96.0	87.4	83.0	**86.1**	4,096
SnapKV	68.0	81.3	24.5	73.4	0
Dense, flagged heads	97.5	85.3	94.0	85.8	3,008 / 3,504
Loki, flagged heads	**98.0**	**88.0**	88.0	84.6	1,216 / 1,264
Repair, +question	96.5	87.4	79.5	84.3	1,216 / 1,264
Repair, every token	97.5	87.8	**90.0**	85.1	1,216 / 1,264

Table 4: **Beyond multi-key retrieval.** RULER means are over 12 tasks, HELMET's real-text mean over nine sets, and LongFuncEval is scored per question. Per-task values in Table 16 and Appendix H.

					Training-free Repair	
Task	Model	Dense	Sparse	Dense, Flagged Heads	Question Tokens	Every Token
RULER	InfLLM-V2-8B, 32K	92.4	87.0	90.4	**90.9**	**90.9**
HELMET, JSON	InfLLM-V2-8B, 32K	92.5	72.0	**93.0**	**93.0**	89.5
HELMET, Real Text	InfLLM-V2-8B, 32K	56.7	56.2	57.1	57.0	**57.6**
LongFuncEval, 32K	InfLLM-V2-8B	62.9	58.4	**61.3**	59.6	60.0
LongFuncEval, 64K	InfLLM-V2-8B	48.3	39.7	44.2	44.2	**46.1**

and on NOSA-8B whose loss is made across the prefill, repair at every token closes more of the gap than the question-token variant. Hence, it is the repair's default. Full attention in the flagged heads closes more still, the most on the retraining whose smaller budget binds. At 64K, twice the base model's native range, neither fix reaches Dense, and both repairs beat the shipped selector given three times as many blocks per token (Table 19). A sweep over prompt lengths and a split of every cell by the needle's depth in the prompt show that the shipped selector loses needles at every depth within the native range and the early positions first beyond it, while full attention in the blind heads returns every band (Figures 5 and 6, Appendix D).

Against other selectors. At the same budget, in every head and at the decode and question tokens, Quest's bound and Double Sparsity's label channels fall below the shipped selector on MK3 of MiniCPM4.1-8B, and SnapKV's selection from the question window recovers part of the loss there (Table 3). Loki's principal directions (Singhania et al., 2024) match the question-token repair on both checkpoints while reading several times its bytes. In the flagged heads alone, keys of a few bits (Ji et al., 2026; Liu et al., 2026b) reach exact mass while mean-pooled keys under the same criterion fall below the shipped selector (Table 13). Loki's directions on flagged heads at every token match the repair on MiniCPM4.1-8B and come close on NOSA-8B (Table 3). This indicates that what matters is token resolution in the right heads, which the 4-bit code keeps cheaply. More budget in every head approaches the repair only at more than twice the blocks per token (Appendix E).

Other tasks. Across RULER, both repair variants raise the mean of InfLLM-V2-8B at 32K (Table 4). On MiniCPM4.1-8B at its native 64K, they match full attention's mean only because the shipped selector beats full attention on variable tracking, while on common word extraction both stay far below it (Tables 3 and 16). The gain is on the multi-key tasks, and at every token it is paid for in part on common word extraction, which the question-token variant avoids. Full attention in flagged heads pays far more on common word extraction and on the variable tracking of

Table 5: **Other deployed selector families**. The per-head criterion on 4-bit keys fills the free slots at each release's own budget and keeps the blocks it always reads. SeerAttention selects at the prefill tokens only and its flagged heads come from the label-free statistic of Appendix C. ‡NSA-8B ships no dense path, so its full attention column selects every block, its repairs act at every prompt token, and its flagged heads are the InfLLM-V2 family's three.

Model	Context	Dense	Sparse	Exact Mass	4-bit Keys, All	4-bit Keys, Flagged
SeerAttention-R	32K					
Qwen3-4B		96.5	18.0	95.5	**96.0**	95.5
Qwen3-8B		99.0	31.5	99.0	**99.5**	**99.5**
Qwen3-14B		100.0	58.5	**100.0**	**100.0**	**100.0**
SeerAttention, Llama-3.1-8B	32K	99.5	71.5	**92.5**	**92.5**	92.0
Dash-8B	16K	99.5	64.5	94.0	**94.5**	74.0
NSA-8B ‡	16K	99.5	13.0	**98.5**	98.0	64.5
	32K	90.0	3.0	**80.0**	78.5	20.5

MiniCPM4.1-8B (Table 16, Appendix E). On HELMET's JSON key-value retrieval (Yen et al., 2024), both fixes close most or all of the deficit, and both leave results at the mean of the real-text sets level, though the repair at every token costs a few points on two question answering sets (Appendix E). On five real-data benchmarks, the one deficit is on identifier retrieval, where the repair recovers part of it at 32K and most of it on the longer split, beyond the base model's native range, and full attention in the flagged heads about as much at 32K and less there (Table 4, Appendix H).

Other selectors. Table 5 applies the scorer to four other selector designs. The SeerAttention-R gate (Gao et al., 2026) loses much of full attention's accuracy on MK3 while exact mass loses nothing, and 4-bit keys recover all of it at every scale, in every head or in the flagged heads alone. The SeerAttention gate (Gao et al., 2024) on Llama-3.1-8B selects only at the prefill tokens, where 4-bit keys match exact mass, itself below full attention. On Dash-8B (Huang et al., 2026), the same three flagged heads carry only about a quarter of the loss, and 4-bit keys in every head recover most of it. NSA-8B, repaired in every head, comes close to every block at 16K (Appendix G). On the InfLLM-V2 selector trained on Qwen3-8B and on Qwen3-4B, the probe flags 11 and 8 of 288 heads respectively, the same 8 on both sizes and from two calibration tasks. Where every line of the haystack is a look-alike, full attention in them recovers more of the loss than full attention in all the other heads together, though not all of it (Figure 1, Appendix C).

Cost. A deployable path keeps the packed 4-bit keys of the flagged heads in a side cache and scans them with Triton kernels (Tillet et al., 2019). It adds a few percent to the time per output token and somewhat more to the time to the first token, where full attention in the flagged heads adds little (Table 17). In the attention kernels alone, the scorer is slower than full attention's extra reads for a few sequences but cheaper with many long ones (Table 18). Full attention in the flagged heads is thus our default. Combining full attention in prefill with the scorer at decode reaches full attention on the three checkpoints we test (Appendix F).

Limitations. The blind heads are few on the InfLLM-V2 family, whose four checkpoints share one backbone and may inherit them (Wu et al., 2024b). The SeerAttention gates need a tenth to a third of their heads, and on Dash-8B and NSA-8B, the family's three heads recover only part of the loss, which is a miss that the label-free statistic of Appendix C finds on Dash-8B. On real API data, the flagged heads carry about half of the loss that exact mass recovers. Calibration needs look-alike distractors and long generations are untested. Token-level selectors are future work (Appendix I).

8 Conclusion

We show that checkpoints of the InfLLM-V2 family lose multi-key retrieval at their shipped budgets mainly in a few blind heads, which attend to the needle while look-alike blocks outscore its pooled window and exact mass keeps it. Full attention in these heads recovers most of the loss at the least cost for a single sequence while a 4-bit scorer does so at the shipped budget.

Reproducibility statement

Every RULER and JSON key-value cell of the main text uses n=200 with stated seeds, except for cells that state a smaller n. Real-text HELMET sets use 45 to 100 prompts and 600 for the three question answering sets that Appendix E reruns. The real-data benchmarks use their own instance counts, stated per cell, and every result file carries per-sample scores, its run settings, the data hash, and the counts of hook calls, rewritten selections, and errors, which a run must match to complete. Every variant is verified by kernel-call counting and by a fidelity check that reproduces the released selections exactly (Appendix B). The calibration probe, the harness, the deployable Triton path with its unit tests, the table and figure generators, and the result files will be released. Every table and every data figure in this paper is generated from the result files by those scripts.

AI use statement

We used an AI assistant (Claude, Anthropic) under the author's direction, and disclose its use in the categories of the ICLR 2027 policy. Writing: it drafted and revised text in every section of the appendix and polished the prose in the main text. The author edited and approved every sentence. Retrieval and discovery: it searched the literature for related and concurrent work, and the author verified every citation against its source. No dataset or prompt was generated by a model. Every reported number was produced by the released code on the stated result files. Every table and figure is generated from those files, and the author takes full responsibility for the content, the claims, and the references.

References

Huzama Ahmad and Se-Young Yun. SpotAttention: Plug-in block-sparse routing for pretrained long-context transformers. *arXiv preprint arXiv:2606.22874*, 2026.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, 2023.

Anthropic. Introducing Claude Opus 5.5. `https://www.anthropic.com/claude-opus-5-5`, September 2026.

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

Amanda Bertsch, Adithya Pratapa, Teruko Mitamura, Graham Neubig, and Matthew R. Gormley. Oolong: Evaluating long context reasoning and aggregation capabilities. *arXiv preprint arXiv:2511.02817*, 2025.

Zhuoming Chen, Ranajoy Sadhukhan, Zihao Ye, Yang Zhou, Jianyu Zhang, Niklas Nolte, Yuandong Tian, Matthijs Douze, Leon Bottou, Zhihao Jia, and Beidi Chen. MagicPIG: LSH sampling for efficient LLM generation. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.16179.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Tri Dao et al. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *arXiv preprint arXiv:2205.14135*, 2022.

DeepSeek-AI. DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025. DSA and its lightning indexer first described in the DeepSeek-V3.2-Exp technical report, September 2025.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S. Kevin Zhou. Ada-KV: Optimizing KV cache eviction by adaptive budget allocation for efficient LLM inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2407.11550.

Tianyu Fu, Haofeng Huang, Xuefei Ning, Genghan Zhang, Boju Chen, Tianqi Wu, Hongyi Wang, Zixiao Huang, Shiyao Li, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. Mixture of attention spans: Optimizing LLM inference efficiency with heterogeneous sliding-window lengths. In *Conference on Language Modeling (COLM)*, 2025a. arXiv:2406.14909.

Yaosheng Fu, Guangxuan Xiao, Xin Dong, Song Han, and Oreste Villa. SparDA: Sparse decoupled attention for efficient long-context LLM inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026. arXiv:2606.04511.

Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. Not all heads matter: A head-level KV cache compression method with integrated retrieval and reasoning. In *International Conference on Learning Representations (ICLR)*, 2025b. arXiv:2410.19258.

Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Peiyuan Zhou, Jiaxing Qi, Junjie Lai, Hayden Kwok-Hay So, Ting Cao, Fan Yang, and Mao Yang. SeerAttention: Learning intrinsic sparse attention in your LLMs. *arXiv preprint arXiv:2410.13276*, 2024.

Yizhao Gao, Shuming Guo, Shijie Cao, Yuqing Xia, Yu Cheng, Lei Wang, Lingxiao Ma, Yutao Sun, Tianzhu Ye, Li Dong, Hayden Kwok-Hay So, Yu Hua, Ting Cao, Fan Yang, and Mao Yang. SeerAttention-R: Sparse attention adaptation for long reasoning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2506.08889.

GLM-5 Team. GLM-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Google DeepMind. Introducing Gemini 3.8 Flash and 3.8 Flash Cyber. `https://blog.google/innovation-and-ai/models-and-research/gemini-models/3-8-flash-and-3-8-flash-cyber/`, September 2026.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Jitai Hao, Yuke Zhu, Tian Wang, Jun Yu, Xin Xin, Bo Zheng, Zhaochun Ren, and Sheng Guo. OmniKV: Dynamic context selection for efficient long-context LLMs. In *International Conference on Learning Representations (ICLR)*, 2025. `https://openreview.net/forum?id=ulCAPXYXfa`.

Coleman Hooper, Sebastian Zhao, Luca Manolache, Sehoon Kim, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Multipole attention for efficient long context reasoning. *arXiv preprint arXiv:2506.13059*, 2025.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Yuxiang Huang, Pengjie Wang, Jicheng Han, Weilin Zhao, Zhou Su, Ao Sun, Hongya Lyu, Hengyu Zhao, Yudong Wang, Chaojun Xiao, Xu Han, and Zhiyuan Liu. NOSA: Native and offloadable sparse attention. *arXiv preprint arXiv:2510.13602*, 2025.

Yuxiang Huang, Nuno M. T. Gonçalves, Federico Alvetreti, Lei Li, Xu Han, Edoardo M. Ponti, André F. T. Martins, and Marcos V. Treviso. Dashattention: Differentiable and adaptive sparse hierarchical attention. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026. arXiv:2605.18753.

Junsung Hwang. Locks: Page-local compact key summaries for efficient long-context decoding. *arXiv preprint arXiv:2607.24555*, 2026.

Xiaodong Ji, Hailin Zhang, Fangcheng Fu, and Bin Cui. Sale : Low-bit estimation for efficient sparse attention in long-context llm prefilling. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2505.24179.

Huiqiang Jiang et al. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. *arXiv preprint arXiv:2407.02490*, 2024.

Shibo Jie, Yehui Tang, Kai Han, Zhi-Hong Deng, and Jing Han. SpeCache: Speculative key-value caching for efficient generation of LLMs. *arXiv preprint arXiv:2503.16163*, 2025.

Vivek Kalyanarangan. Fathom: Per-query read depth for sparse decoding over offloaded kv caches. *arXiv preprint arXiv:2609.17652*, 2026.

Kiran Kate, Tejaswini Pedapati, Kinjal Basu, Yara Rizk, Vijil Chenthamarakshan, Subhajit Chaudhury, Mayank Agarwal, and Ibrahim Abdelaziz. LongFuncEval: Measuring the effectiveness of long context models for function calling. *arXiv preprint arXiv:2505.10570*, 2025.

Kimi Team. Kimi K3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Xunhao Lai, Weiqi Xu, Yufeng Yang, Qiaorui Chen, Yang Xu, Lunbin Zeng, Xiaolong Li, Haohai Sun, Haichao Zhu, Vito Zhang, Jinkai Hu, Jiayao Li, et al. Minimax sparse attention. *arXiv preprint arXiv:2606.13392*, 2026.

Wonbeom Lee, Jungi Lee, Junghwan Seo, and Jaewoong Sim. InfiniGen: Efficient generative inference of large language models with dynamic KV cache management. In *USENIX Symposium on Operating Systems Design and Implementation*, 2024.

Jiaqi Leng, Xiang Hu, Junxiong Wang, Jianguo Li, Wei Wu, and Yucheng Lu. Understanding and improving length generalization in hierarchical sparse attention models. In *International Conference on Learning Representations*, 2026. arXiv:2510.17196.

Jia Li, Xuyuan Guo, Lei Li, Kechi Zhang, Ge Li, Jia Li, Zhengwei Tao, Fang Liu, Chongyang Tao, Yuqi Zhu, and Zhi Jin. LongCodeU: Benchmarking long-context language models on long code understanding. *arXiv preprint arXiv:2503.04359*, 2025.

Yuhong Li et al. SnapKV: LLM knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*, 2024.

Opher Lieber et al. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

Chaofan Lin, Jiaming Tang, Shuo Yang, Hanshuo Wang, Tian Tang, Boyu Tian, Ion Stoica, Song Han, and Mingyu Gao. Twilight: Adaptive attention sparsity with hierarchical top-p pruning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2502.02770.

Yuping Lin, Zitao Li, Yue Xing, Pengfei He, Yingqian Cui, Yaliang Li, Bolin Ding, Jingren Zhou, and Jiliang Tang. Retrieval heads are dynamic. *arXiv preprint arXiv:2602.11162*, 2026.

Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, Chen Chen, Fan Yang, Yuqing Yang, and Lili Qiu. RetrievalAttention: Accelerating long-context LLM inference via vector retrieval. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2409.10516.

Di Liu, Ruitian Wang, Chen Chen, Mingliang Gong, Yongjie Yuan, Han Zhao, Yu Feng, Quan Chen, and Minyi Guo. AB-Sparse: Sparse attention with adaptive block size for accurate and efficient long-context inference. *arXiv preprint arXiv:2605.12110*, 2026a.

Jiawei Liu, Jia Le Tian, Vijay Daita, Yuxiang Wei, Yifeng Ding, Yuhan Katherine Wang, Jun Yang, and Lingming Zhang. RepoQA: Evaluating long context code understanding. *arXiv preprint arXiv:2406.06025*, 2024a.

Zhigeng Liu, Zhiyuan Ning, Ruixiao Li, Xiaoran Liu, Yuerong Song, Min Zhang, Ziwei He, and Xipeng Qiu. Faster than flash: Exploiting attention sparsity for efficient long-context decoding. *arXiv preprint arXiv:2609.00097*, 2026b.

Zirui Liu et al. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. *arXiv preprint arXiv:2402.02750*, 2024b.

Enzhe Lu et al. MoBA: Mixture of block attention for long-context LLMs. *arXiv preprint arXiv:2502.13189*, 2025.

Joel Max. Quantizing for minimum distortion. *IRE Transactions on Information Theory*, 6(1):7–12, 1960.

MiniCPM Team. MiniCPM4: Ultra-efficient LLMs on end devices. *arXiv preprint arXiv:2506.07900*, 2025.

OpenAI. OpenAI MRCR: Long context multiple needle in a haystack benchmark. `https://huggingface.co/datasets/openai/mrcr`, 2025. MIT licence; revision f4c69fa.

OpenAI. GPT-6 Astra: A new generation of intelligence. `https://openai.com/index/gpt-6-astra/`, 2026.

Xingyu Ren, Youran Sun, Chugang Yi, and Haizhao Yang. Understanding sparse attention selectivity in long-context foundation models via counterfactual evaluation. *arXiv preprint arXiv:2608.01676*, 2026.

Jay Shah et al. FlashAttention-3: Fast and accurate attention with asynchrony and low-precision. *arXiv preprint arXiv:2407.08608*, 2024.

Prajwal Singhania, Siddharth Singh, Shwai He, Soheil Feizi, and Abhinav Bhatele. Loki: Low-rank keys for efficient sparse attention. In *Advances in Neural Information Processing Systems*, 2024.

Chuxu Song, Jiuqi Wei, and Zhencan Peng. RBS-Attention: Radius-bounded sparse prefill for long-context large language models. *arXiv preprint arXiv:2609.20971*, 2026.

Hanshi Sun, Li-Wen Chang, Wenlei Bao, Size Zheng, Ningxin Zheng, Xin Liu, Harry Dong, Yuejie Chi, and Beidi Chen. ShadowKV: KV cache in shadows for high-throughput long-context LLM inference. In *International Conference on Machine Learning (ICML)*, 2025a. arXiv:2410.21465.

Yutao Sun, Zhenyu Li, Yike Zhang, Tengyu Pan, Bowen Dong, Yuyi Guo, and Jianyong Wang. Efficient attention mechanisms for large language models: A survey. *arXiv preprint arXiv:2507.19595*, 2025b.

Hanlin Tang, Yang Lin, Jing Lin, Qingsen Han, Shikuan Hong, Yiwu Yao, and Gongyi Wang. RazorAttention: Efficient KV cache compression through retrieval heads. *arXiv preprint arXiv:2407.15891*, 2024a.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context LLM inference. *arXiv preprint arXiv:2406.10774*, 2024b.

Zecheng Tang, Quantong Qiu, Yi Yang, Zhiyi Hong, Haiya Xiang, Kebin Liu, Qingqing Dang, Juntao Li, and Min Zhang. Elastic attention: Test-time adaptive sparsity ratios for efficient transformers. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2601.17367.

Alexander Tian, Aditya Ghai, Sanjit Neelam, Zaal Vasania, and Akshay Mishra. COBS: Cumulant order block sparse attention. *arXiv preprint arXiv:2607.09052*, 2026.

Philippe Tillet, Hsiang-Tsung Kung, and David Cox. Triton: An intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages (MAPL)*, 2019.

Dongwei Wang, Zijie Liu, Song Wang, Yuxin Ren, Jianing Deng, Jingtong Hu, Tianlong Chen, and Huanrui Yang. FIER: Fine-grained and efficient KV cache retrieval for long-context LLM inference. *arXiv preprint arXiv:2508.08256*, 2025.

Hongxing Wang, Harenome Razanajato, Zhen Zhang, Yujie Yuan, and Hongsheng Liu. How much dense attention is necessary? Oracle-Guided sparse prefill for full/GQA layers in hybrid long-context models. *arXiv preprint arXiv:2606.07703*, 2026a.

Xinghao Wang, Pengyu Wang, Xiaoran Liu, Fangxu Liu, Jason Chu, Kai Song, and Xipeng Qiu. Prism: Spectral-aware block-sparse attention. In *International Conference on Machine Learning (ICML)*, 2026b. arXiv:2602.08426.

Thomas Wolf, Lysandre Debut, Victor Sanh, et al. HuggingFace's transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020. arXiv:1910.03771.

Wei Wu, Zhuoshi Pan, Chao Wang, Liyi Chen, Yunchu Bai, Tianfu Wang, Kun Fu, Zheng Wang, and Hui Xiong. Tokenselect: Efficient long-context inference and length extrapolation for llms via dynamic token-level kv cache selection. *arXiv preprint arXiv:2411.02886*, 2024a.

Wenhao Wu et al. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*, 2024b.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Guangxuan Xiao, Junxian Guo, Kasra Mazaheri, and Song Han. Optimizing mixture of block attention. *arXiv preprint arXiv:2511.11571*, 2025.

Guangxuan Xiao et al. DuoAttention: Efficient long-context LLM inference with retrieval and streaming heads. *arXiv preprint arXiv:2410.10819*, 2024.

Di Xiu, Hongyin Tang, Bolin Rong, Lizhi Yan, Jingang Wang, Yifan Lu, and Xunliang Cai. A preliminary study on the promises and challenges of native top-k sparse attention. *arXiv preprint arXiv:2512.03494*, 2025.

Wenyan Xu and Alizer Wong. Bmfa: Boundary-minority free-energy adaptive screening. *arXiv preprint arXiv:2607.17656*, 2026.

Yufei Xu, Fanxu Meng, Fan Jiang, Yuxuan Wang, Ruijie Zhou, Zhaohui Wang, Jiexi Wu, Zhixin Pan, Xiaojuan Tang, Wenjie Pei, Tongxuan Liu, Di Yin, et al. Hisa: Efficient hierarchical indexing for fine-grained sparse attention. *arXiv preprint arXiv:2603.28458*, 2026.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Lijie Yang, Zhihao Zhang, Zhuofu Chen, Zikun Li, and Zhihao Jia. TidalDecode: Fast and accurate LLM decoding with position persistent sparse attention. In *International Conference on Learning Representations (ICLR)*, 2025b. arXiv:2410.05076.

Shang Yang, Junxian Guo, Haotian Tang, Qinghao Hu, Guangxuan Xiao, Jiaming Tang, Yujun Lin, Zhijian Liu, Yao Lu, and Song Han. LServe: Efficient long-sequence LLM serving with unified sparse attention. In *Proceedings of Machine Learning and Systems (MLSys)*, 2025c. arXiv:2502.14866.

Shuo Yang, Ying Sheng, Joseph E. Gonzalez, Ion Stoica, and Lianmin Zheng. Post-training sparse attention with double sparsity. *arXiv preprint arXiv:2408.07092*, 2024.

Xu Yang, Jiapeng Zhang, Zhangke, Changjian Chen, Yuxin Chen, Feiqiang Sun, Chengguang Xu, Feng Jin, and Zhuo Tang. Self-indexing attention for compression-compatible sparse long-context llm inference. *arXiv preprint arXiv:2609.13205*, 2026.

Howard Yen et al. HELMET: How to evaluate long-context language models effectively and thoroughly. *arXiv preprint arXiv:2410.02694*, 2024.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Y. X. Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.

Wen Zan, Jiaqi Zhang, Jianchao Tan, Hong Liu, Cunguang Wang, Xiang Li, et al. Longcat sparse attention: Taming the lightning via streaming-aware hierarchical cross-layer indexing. *arXiv preprint arXiv:2608.01662*, 2026.

Hailin Zhang, Xiaodong Ji, Yilin Chen, Fangcheng Fu, Xupeng Miao, Xiaonan Nie, Weipeng Chen, and Bin Cui. PQCache: Product quantization-based KVCache for long context LLM inference. *Proceedings of the ACM on Management of Data*, 3(3):201:1–201:30, 2025. doi: 10.1145/3725338. SIGMOD 2025; arXiv:2407.12820.

Zhenyu Zhang et al. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. *arXiv preprint arXiv:2306.14048*, 2023.

Weilin Zhao, Zihan Zhou, Zhou Su, Chaojun Xiao, Yuxuan Li, Yanghao Li, Yudi Zhang, Weilun Zhao, Zhen Li, Yuxiang Huang, Ao Sun, Xu Han, et al. InfLLM-V2: Dense-sparse switchable attention for seamless short-to-long adaptation. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.24663.

Zhipu AI. GLM-5.3: Frontier coding with emergent cyber capabilities. `https://z.ai/blog/glm-5.3`, 2026.

Yanke Zhou, Yiduo Li, Hanlin Tang, Maohua Li, Kan Liu, Tao Lan, Lin Qu, Yuan Yao, and Xiaoxing Ma. Full attention strikes back: Transferring full attention into sparse within hundred training steps. *arXiv preprint arXiv:2605.16928*, 2026.

Guide to the appendix. Appendix A gives the checkpoints and the protocol, Appendix B the harness, its controls and the five draws, Appendix C the calibration probe, Appendix D the theory and its tests, Appendix E the ablations, per-task results and baselines, Appendix F the measured cost, Appendix G the other checkpoints, Appendix H real data, and Appendix I the comparison class and the closest work.

A Checkpoints and evaluation protocol

Checkpoints. The four InfLLM-V2 checkpoints share the MiniCPM4-8B backbone, 32 layers of 32 query heads and 2 key-value heads of dimension 128, 16 query heads per group. Their selector scores keys mean-pooled over windows of 32 at a stride of 16 and maps the window scores onto blocks of 64 tokens. It always reads the initial block, a local window of 2048 tokens and the query's own block. There are 34 forced blocks total, and the remaining 62 free slots are filled by Equation 1, 96 blocks per token. Tokens below position 8192 attend densely. The retraining's kernel counts a local window of 1024 tokens inside its top-k and reads 64 blocks, 18 forced and 46 free, with no dense threshold. NOSA-8B has the same shape and tokenizer as the MiniCPM4-8B backbone and was trained by its authors, whose release does not state its initialization. Each checkpoint's full attention setting loads the same weights with the release's sparse configuration removed. The other families are described where they are evaluated.

Protocol. We use RULER (Hsieh et al., 2024) at a pinned generator commit, with templates verified byte-identical to each checkpoint's own, canonical generation caps, and chain-of-thought disabled. A MK3 prompt is a haystack of lines that pair a UUID key with a UUID value, one of which holds the queried key. The haystack is followed by the question, the chat template and RULER's answer prefix, 86 to 104 tokens total. Every InfLLM-V2 checkpoint and NOSA-8B read the same prompts, which average 31,887 tokens at 32K and 64,829 at 64K. Generation is capped at 128 tokens. The answers of Table 2 average 32 to 38 generated tokens, so the decode tokens are a few dozen steps. The last 128 prompt tokens hold the question, the template, the answer prefix and the end of the last line. Every RULER prompt draws its own keys, values and needle position. Every RULER cell reports string-match accuracy on the same draw of 200 prompts, or on its first 100, 50 or 25 prompts where stated, apart from the four further draws of Table 6. Each difference is tested with a paired bootstrap, 10000 resamples of the prompts with a fixed seed, and the tables report point values. The 95 percent intervals exclude zero for every gain of the repair and of the full attention in the flagged heads over the shipped selector that the text reports, on MK3, on HELMET's JSON key-value retrieval and on LongFuncEval's identifier questions. They also exclude zero for two against three flagged heads. They include zero for every difference the text calls level. Among them are the two repair variants on the held-out draws, the repair against full attention in the flagged heads on InfLLM-V2-8B and MiniCPM4.1-8B, two, three and sixteen bits against four, the group-mean query against the per-head criterion, the scorer in the other head of the flagged layers, full attention in the unflagged heads against the shipped selector on InfLLM-V2-8B and on MiniCPM4.1-8B at

64K, and both repair variants on HELMET's real-text sets. Two identical shipped runs on different GPU types differed by 1.5 points, so the text interprets no difference of that size. A paired test on one draw decides the larger ones. Every result file records the GPU types it ran on. The calibration draw shares no prompt with the evaluation draw.

B HARNESS FIDELITY, RUN GATES AND FIVE DRAWS

Fidelity. All arms act through a hook on the selector's output. A keep-only arm that retains the forced blocks and fills the free slots from the kernel's own score reproduces the released InfLLM-V2 selector exactly, 0 mismatches in 42.9 million slots at n=200. It scores 65.5, identical to the shipped arm. Listing both heads of a layer reproduces the layer-level arm token for token. Rewriting the last prefill tokens inside the single prefill pass scores 95.5 with exact mass, equal to a two-pass protocol through the decode path. Every result file records the hook calls, rewritten selections, decode forwards and errors. A run aborts unless they equal their expected values, 19,132,275 rewritten selections for three heads on the 200 prompts of MK3 at 32K. On SeerAttention-R, the override fills exactly as many free blocks as the gate selects, and the Triton path is checked against the emulation before any timing. Its decode picks differ from the emulation's in 2.4 percent of the checked slots at 32K and 3.3 percent at 64K, above the check's 3 percent threshold. The scorer's 64K rows of Table 17 therefore time a path whose selections were not confirmed at that length.

Negative controls. Six controls run through the same code path as the repair (Table 13). The per-head scorer on 4-bit keys at every token of the *other* key-value head of layers 9, 16, and 22 rewrites the same 19,132,275 selections and scores 64.5, -1.0 against the shipped selector. Random free blocks at the decode tokens of the three flagged heads score 12.5, and random free blocks in the other head of those layers score 64.5. A fourth control keeps the heads, the tokens and the per-head criterion. It replaces the 4-bit keys by mean-pooled keys, one per 64 tokens, and scores 53.5. A fifth, run on an H100, puts the scorer at every token of the 61 unflagged heads and leaves the flagged three heads to the shipped selector. It scores 66.5 against 67.0 for the shipped selector, a difference the paired test does not separate from zero. A sixth gives full attention to three heads drawn at random from the 61 unflagged, in three seeds. It scores 66.0, 63.5 and 66.5 against the same 67.0, none above it. The gain is therefore specific to the flagged heads, to the blocks the scorer picks, and to token resolution. It does not come from rewriting selections or from full attention in any three heads.

Five evaluation draws. Every other RULER cell of this paper uses one draw of 200 prompts per task and length, or its first 100, 50 or 25 prompts where stated. For the three settings of Table 2, we repeat the shipped selector, full attention in the flagged heads, both repair variants and full attention on five draws from five generator seeds (Table 6). The heads are flagged on the calibration draw of seed 7, and the bit width follows the probe of Appendix D on that draw. The rule's form and threshold, the token set, the quantizer's prefix and the every-token default were settled while watching the first evaluation draw, the repair's best draw on InfLLM-V2-8B and the only one where it reaches full attention. The other four draws were generated after these choices, and the abstract quotes their means. On those means, the question-token variant loses nothing to the every-token default, and full attention in the flagged heads gains within a point of it. The ranges of Table 6 are spreads of absolute accuracy across draws. Per draw, the repair at every token stays within 5.0, 0.0 and 2.5 points of full attention on the three settings, and full attention in the flagged heads stays within 3.0, 0.0 and 1.5.

Table 6: **Five evaluation draws** of 200 MK3 prompts each, from five seeds of RULER's generator. Each entry is the mean over the 1000 prompts and, in brackets, the range over the five draws. The lower block gives the same over the 800 prompts of the four draws held out from every design choice, which the abstract quotes. "Dense, Flagged Heads" gives the flagged heads full attention at every token, "+ Question" repairs the decode tokens and the last 128 prompt tokens, and "Every Token" repairs every token, all in the heads that the checkpoint's own calibration flags.

Model	Context	Dense	Sparse	Dense, Flagged Heads	+ Question	Every Token
InfLLM-V2-8B	32K	98.0 (97.0 to 98.5)	64.8 (62.0 to 70.5)	**96.4 (94.5 to 98.0)**	94.8 (93.0 to 96.0)	95.4 (93.5 to 98.0)
MiniCPM4.1-8B	32K	100.0 (100.0 to 100.0)	83.2 (81.5 to 85.0)	**100.0 (100.0 to 100.0)**	99.8 (99.5 to 100.0)	**100.0 (100.0 to 100.0)**
MiniCPM4.1-8B	64K	97.3 (96.0 to 98.5)	55.6 (48.5 to 64.5)	96.8 (95.0 to 97.5)	**96.9 (96.5 to 98.0)**	**96.9 (94.0 to 98.0)**
Draws two to five, held out from every design choice, 800 prompts						
InfLLM-V2-8B	32K	97.9 (97.0 to 98.5)	64.6 (62.0 to 70.5)	**96.0 (94.5 to 97.5)**	95.0 (93.0 to 96.0)	94.8 (93.5 to 95.5)
MiniCPM4.1-8B	32K	100.0 (100.0 to 100.0)	83.5 (81.5 to 85.0)	**100.0 (100.0 to 100.0)**	99.9 (99.5 to 100.0)	**100.0 (100.0 to 100.0)**
MiniCPM4.1-8B	64K	97.4 (96.0 to 98.5)	55.6 (48.5 to 64.5)	96.6 (95.0 to 97.5)	**97.0 (96.5 to 98.0)**	96.8 (94.0 to 98.0)

C CALIBRATION AND THE FLAGGING RULE

The rule. The probe runs the shipped model with its own selector and teacher-forces the answer tokens after the prompt's sparse prefill. It reads every quantity below from the queries and cached keys of that run. At decode step t, the model must produce the needle token at context position a. The needle-containing block has index $n = \lfloor a/L \rfloor$, for block length L. For a key-value head h, let $\mathcal{P}_h$ collect these pairs (t, n) over all calibration prompts whose needle block is not forced. Let $\mathcal{S}_t^h$ be the head's shipped selection at token t, and let $\mathcal{E}_t^h = \mathrm{TopK}_{b \notin \mathcal{F}}(M_{t,b}^h,\, K)$ be the selection that exact mass makes at the same budget, with $M_{t,b}^h$ being the head's exact mass from Equation 2. The probe records the needle recall of a selection $\mathcal{A}$ and the head's mean needle mass,

$$r_h(\mathcal{A}) = \frac{1}{|\mathcal{P}_h|} \sum_{(t,n)\in\mathcal{P}_h} \mathbf{1}\big[n \in \mathcal{A}_t^h\big], \qquad m_h = \frac{1}{|\mathcal{P}_h|} \sum_{(t,n)\in\mathcal{P}_h} \frac{M_{t,n}^h}{G}, \tag{6}$$

and flags the heads

$$\mathcal{H} = \big\{h \,:\, \big(r_h(\mathcal{E}) - r_h(\mathcal{S})\big)\, m_h > \tau\big\}. \tag{7}$$

In words, for every key-value head, the probe records the recall of the needle block under the shipped selector and under exact mass at the same free budget, and the mean attention mass the head's group puts on the needle block. It flags the head when the recall gap times the mass exceeds 0.01, a product of means that approximates one percent of the attention mass that the shipped selection forfeits. Table 7 lists the flagged heads, and Figure 1 draws every head. The flagged heads of layers 9, 16, and 22 score 0.014 to 0.058, the fourth head of MiniCPM4.1-8B scores 0.010, and the best unflagged head scores at most 0.009. Early layers have recall gaps above 0.3 but put under half a percent of their attention on the needle, and late layers read it and already find it. Both factors are needed, since the three heads with the largest recall gap alone give 82.0 end to end and the three with the largest needle mass alone give 80.5, against 98.0 for the product (Table 13).

A general form. The rule is one instance of a per-head score

$$S_h = c_h \left(f_h(\mathcal{A}_{\mathrm{ref}}) - f_h(\mathcal{A}_{\mathrm{ship}})\right)^p, \qquad \mathcal{H} = \{h : S_h > \tau\}, \tag{8}$$

where $f_h(\mathcal{A})$ measures over a calibration set how well a selection $\mathcal{A}$ of the head's blocks supports the behavior of interest, and $\mathcal{A}_{\mathrm{ref}}$ is a reference selection. The factor c_h weights the head by its stake in that behavior, read from its queries, keys and values. $p \geq 1$ sets how much a large gap counts against many small ones. The rule we adopt in our calibration takes f_h as needle recall, $\mathcal{A}_{\mathrm{ref}}$ as exact mass at the shipped budget so that the gap charges the selector and not the budget, c_h as the mean needle mass, and $p = 1$ where the product of means approximates the needle mass the shipped selection forfeits. The label-free statistic below is the instance with f_h the captured mass, $c_h = 1$ and no needle. A retrieval-head score (Wu et al., 2024b) is the stake c_h alone with no gap, which is why it ranks heads whose selector already finds the needle above the flagged ones. Full attention as the reference would add the budget's own cost to the gap, and $p > 1$ would favor a few heads with large gaps. Whatever the instance, the threshold is set once and the flagged set is checked end to end.

A scan without the probe. Figure 3 gives each of the 64 key-value heads of InfLLM-V2-8B full attention alone, with the other 63 keeping the shipped selector, on the evaluation draw at 32K, and takes its gain against the shipped selector on the same GPU type. The three flagged heads gain 15.5, 23.0 and 17.0 points in layers 9, 16 and 22, and no other head gains more than 2.0, with a median of 0.0. Intervening on one head at a time, with no probe, thus singles out the same three heads. The heads that the retrieval score below ranks first, whose selector already finds the needle, are the second head of layer 29 and the first of layer 26, which gain -0.5 and 0.0. We do not run DuoAttention's gate optimization, which learns a mix of full and local attention per head, since this scan already measures what full attention in each head gains, the quantity any head ranking has to match.

Retrieval heads and blind heads overlap in part. On the same calibration prompts and teacher-forced tokens, the retrieval score of Wu et al. (2024b), the fraction of needle tokens on which a query head's most-attended key is the token being copied, ranks the second head of layer 29 first at 0.71. The first heads of layers 22 and 26 follow at 0.67 and 0.66. The flagged head of layer 9 comes fifth at 0.33, and the flagged head of layer 16 comes fifteenth at 0.02. In the heads of layers 26 and 29, the shipped selector already finds the needle block at 98 percent of the probed tokens, so the

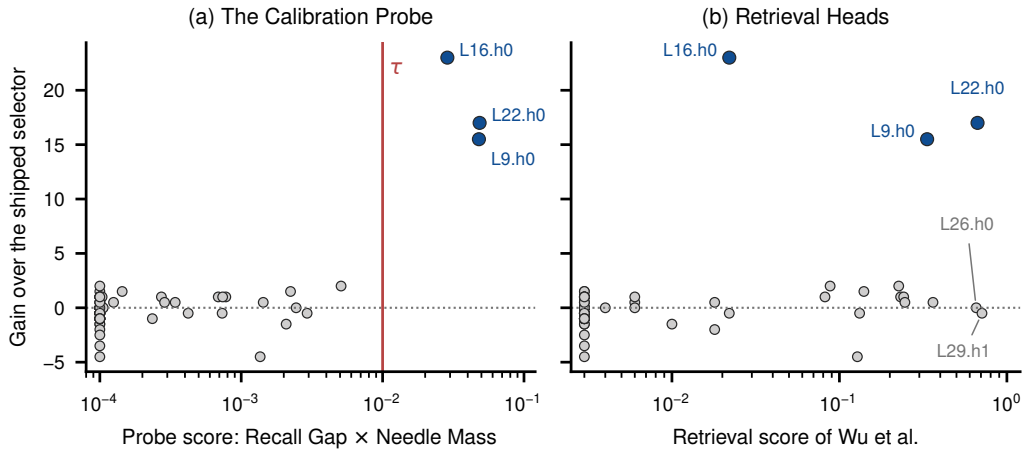


Figure 3: **A scan without the probe** (InfLLM-V2-8B, MK3 at 32K, 200 prompts per head). Each point is one of the 64 key-value heads given full attention alone, the other 63 keeping the shipped selector, plotted as its gain over the shipped selector on the same draw and GPU type against (a) the calibration probe's score, with the threshold τ, and (b) the retrieval score of Wu et al. (2024b), both on the calibration draw. The flagged heads are dark.

score favors the heads that read the needle whether or not their selector misses it. The score's three highest heads hold one of the three flagged ones. End to end, the repair in these three heads scores 81.0 and full attention in them scores 82.5, about what full attention in the flagged head of layer 22 gives alone, against 98.0 for the repair in the three flagged heads (Table 13). Full attention in the five highest heads, which hold two of the flagged ones, scores 95.5. Full attention in the fifteen highest, the first set that holds all three, scores 98.5 against 98.0 in the three flagged heads alone. Thus, a retrieval score recovers the loss once its set contains the flagged heads, but at five times as many heads.

Table 7: **Flagged key-value heads** (50 calibration prompts, MK3, 32K, 10 teacher-forced answer tokens each). For each flagged head, recall of the needle block under the shipped selector / under exact mass / mean needle mass, and in parentheses the flagging score. The last column is the largest score of an unflagged head.

Checkpoint	Flagged Heads	Best Unflagged
InfLLM-V2-8B	L9.h0 0.19 / 0.79 / 0.080 (0.048), L16.h0 0.24 / 0.76 / 0.056 (0.029), L22.h0 0.73 / 0.99 / 0.192 (0.049)	0.0051
MiniCPM4-8B Instruct	L9.h0 0.16 / 0.77 / 0.080 (0.048), L16.h0 0.26 / 0.77 / 0.053 (0.027), L22.h0 0.79 / 0.97 / 0.197 (0.037)	0.0072
InfLLMv2-8B Retraining	L9.h0 0.14 / 0.76 / 0.068 (0.042), L16.h0 0.16 / 0.67 / 0.029 (0.014), L22.h0 0.48 / 0.90 / 0.117 (0.049)	0.0086
MiniCPM4.1-8B	L9.h0 0.17 / 0.79 / 0.094 (0.058), L16.h0 0.24 / 0.77 / 0.068 (0.036), L22.h0 0.80 / 0.97 / 0.225 (0.038), L23.h0 0.82 / 0.92 / 0.106 (0.010)	0.0046

Seed, size, length and source. Every evaluated cell uses heads flagged on 50 prompts drawn at seed 7, whose needles occur in no evaluation draw, ten teacher-forced answer tokens each. Five generator seeds return the same three heads, with the smallest flagged score 0.026 to 0.030 and the largest unflagged 0.004 to 0.009. The per-head scores of the five MiniCPM4-8B checkpoints correlate at 0.89 to 0.99. A random half of a draw returns the deployed set in 99 to 100 percent of 2000 subsets on four seeds and 76 percent on the fifth. A draw at 16K returns the same heads. A draw at 64K adds the second head of layer 29, with which the repair scores 98.5 at 32K against 98.0 and 69.0 at 64K against 66.0. The haystack decides which heads a source can reveal (Table 8). RULER's needle haystacks and HELMET's JSON key-value data, on examples HELMET does not evaluate, flag the heads of layers 9, 16, and 22, a subset of them or one more, and their sets score 95.0 to 98.5 end to end. In contrast, essay haystacks and DuoAttention's released multi-passkey items (Xiao et al., 2024) flag the head of layer 9 with early heads, because there the shipped selector finds the needle in layers 16 and 22. Their sets score 79.5 to 84.5. A pooled score misranks a needle only against competitors whose mean logit is elevated (Proposition 1), which look-alike needles supply

and book text does not, so a calibration set must contain look-alike competitors. Every such set we tried recovers at least 29.5 of the 33 points.

Table 8: **Calibration-set ablation on InfLLM-V2-8B.** Every row reruns the flagging rule with the default threshold on a different calibration set. Columns give the number of prompts, the flagged key-value heads, the smallest flagged and the largest unflagged score, the probability that a random subset of 10 or 25 of the row's prompts returns the deployed set of three heads (2000 subsets without replacement), and the accuracy on MK3 at 32K of the every-token repair run with the row's flagged set ($n = 200$, shipped selector 65.5, full attention 98.5). A dash is a quantity the row cannot have, a smallest flagged score where the rule flags nothing, or a subset probability at a k larger than the row's own number of prompts.

			Score		P(Deployed Set)		
Calibration Set	Prompts	Flagged Heads	Flagged	Unflagged	k=10	k=25	Repair
Random seed of RULER's generator (MK3, 32K)							
Seed 7, the default	50	L9.h0, L16.h0, L22.h0	0.029	0.005	0.91	1.00	98.0
Seed 11	50	L9.h0, L16.h0, L22.h0	0.027	0.006	0.81	0.99	98.0
Seed 13	50	L9.h0, L16.h0, L22.h0	0.030	0.007	0.83	0.99	98.0
Seed 17	50	L9.h0, L16.h0, L22.h0	0.026	0.004	0.99	1.00	98.0
Seed 19	50	L9.h0, L16.h0, L22.h0	0.026	0.009	0.67	0.76	98.0
Context length (MK3, seed 7)							
16K	50	L9.h0, L16.h0, L22.h0	0.018	0.005	0.62	0.91	98.0
64K	50	L9.h0, L16.h0, L22.h0, L29.h1	0.012	0.007	0.33	0.18	98.5
Another RULER task as the source (32K, seed 7)							
MK2	50	L9.h0, L22.h0	0.021	0.009	0.15	0.09	95.0
MK1	50	L3.h0, L6.h0, L9.h0	0.011	0.010	0.00	0.00	82.0
Multi-value	50	L6.h0, L6.h1, L9.h0, L29.h1	0.013	0.009	0.00	0.00	79.5
Single 1	50	none	–	0.003	0.00	0.00	65.5
Single 3	50	L6.h0, L9.h0	0.011	0.010	0.00	0.00	84.0
A source outside RULER (32K)							
HELMET JSON KV, held-out examples	50	L9.h0, L16.h0, L22.h0	0.025	0.008	0.53	0.83	98.0
DuoAttention passkeys, 32 words	10	L9.h0	0.026	0.006	–	–	84.0
DuoAttention passkeys, 4 words	10	L3.h0, L9.h0	0.013	0.006	–	–	84.5

The threshold. Any threshold between a checkpoint's largest unflagged score and its smallest flagged score returns the same set. On the four InfLLM-V2 checkpoints and Dash-8B, the smallest flagged score is 1.7 to 5.6 times the largest unflagged one. On MiniCPM4.1-8B, the fourth flagged head scores 0.0102, just above the default threshold. On SeerAttention-R, the probe flags 57, 66 and 69 of the key-value heads of Qwen3-4B, 8B and 14B. The scores form a continuum, and the outcome is insensitive above a floor, since the 8B's 32 highest-scoring heads, the 66 the threshold returns and all 288 score 99.5 alike. On InfLLM-V2-8B, two heads give 95.0, the default three give 98.0, and ten give 99.0, against 99.5 with the scorer in all 64 heads. On MiniCPM4.1-8B, the question-token variant scores the same with its three common heads or four.

A label-free statistic. The captured-mass shortfall, the attention mass that exact selection captures in the free slots of a head minus the mass that the shipped selection captures, needs no needle position. On InfLLM-V2-8B, it is 0.144, 0.138, and 0.118 in the flagged heads and at most 0.057 elsewhere. On Dash-8B, it exceeds 0.08 in 54 of 64 heads, where the scorer in those 54 heads scores 94.0, level with 94.5 in every head and far above the 74.0 of the probe's three heads (Table 5). The statistic thus flags the spread-out loss that the needle probe misses there. SeerAttention reads every block at decode, which leaves the needle probe nothing to test. Its heads therefore come from this statistic per query head over the prefill tokens of a calibration draw that shares no answer with the evaluation draw. At the same threshold $\tau = 0.01$, it flags 365 of the 1024 query heads, and the repair in them scores 92.0 against 92.5 in every head (Table 5).

A second backbone. Tang et al. (2026) release InfLLM-V2 checkpoints continued from Qwen3-8B, for 1000 steps, and from Qwen3-4B, each with 288 key-value heads in 36 layers. With full attention, they score 7.5 and 1.0 on MK3 at 32K. We therefore test them at 32K over 200 prompts on MK2 where every line of the haystack is a look-alike, and on MK1 which hides the needle and three look-alikes in an essay (Table 9). Calibrated on 50 prompts of a disjoint MK3 draw at the default threshold, the probe flags 11 heads of the 8B, in layers 7 to 24, and 8 of the 4B, all among the 8B's 11. Calibrated on a disjoint MK2 draw instead, it flags the same 8 heads on both sizes, the 8B's three further heads being the lowest ranked of its 11. The set is thus stable across calibration tasks and model sizes, as it is across the InfLLM-V2 checkpoints. On MK2, the 8B loses 35.5 points. Full attention in its 11 flagged heads recovers 20.0, and the repair in them recovers 18.0. Full attention

in the other 277 heads, with the flagged ones left to the shipped selector, recovers 8.5, significantly less than the 11 heads. On MK1, full attention in either set recovers most of the 17.0 points lost. On the 4B, the flagged and the other heads recover 9.0 and 6.5 of 23.5 points on MK2 and 15.0 and 12.0 of 20.0 on MK1, differences the paired test does not separate. The probe's few heads thus matter far more per head than the rest, recovering about as much of the loss as all the other heads together. On this backbone, however, the loss is not confined to them as it is on the InfLLM-V2 family. A rerun of the 8B's first 100 prompts on an A100 gives the same ordering.

Table 9: **A second backbone** (32K). The InfLLM-V2 selector continued from Qwen3-8B and from Qwen3-4B (Tang et al., 2026), each with its own MK3 calibration probe, which flags 11 and 8 of their 288 key-value heads. Full attention in the flagged heads keeps the shipped selection in the others, and full attention in the other heads leaves the flagged ones to the shipped selector. The repair is the default of Section 6 in the flagged heads. The cells of a row ran on one GPU type over the first prompts of the draw.

Checkpoint	GPU	Prompts	Task	Dense	Sparse	Dense, Flagged Heads	Dense, Other Heads	Repair, Flagged Heads
Qwen3-8B, 11 flagged	H100	200	MK1	70.5	53.5	65.0	**67.5**	61.5
			MK2	55.0	19.5	**39.5**	28.0	37.5
Qwen3-8B, 11 flagged	A100	100	MK1	67.0	48.0	62.0	**68.0**	63.0
			MK2	54.0	20.0	**40.0**	22.0	38.0
Qwen3-4B, 8 flagged	A100	200	MK1	57.0	37.0	**52.0**	49.0	51.0
			MK2	31.5	8.0	**17.0**	14.5	13.5

How many heads. Figure 4 gives full attention to the K heads that the 8B's probe scores highest on MK2, the others keeping the shipped selection. Doing so recovers 21.0 of the 35.5 points with the five heads above $\tau = 0.03$, about as many as the 11 at the default threshold. More heads in the probe's order add slowly, 27.0 points with 32 heads and 28.0 with 96. In contrast, 11, 23 and 60 heads drawn at random recover 1.5, 0.5 and -0.5, none of them separated from zero by the paired test. The probe thus ranks the heads by how much of the loss they carry, and a handful carry most of it. Yet 144 heads, half of them, still leave a quarter of the loss.

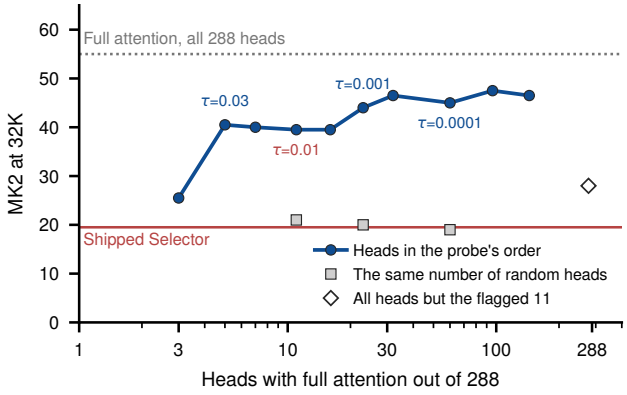


Figure 4: **Heads given full attention** on the InfLLM-V2 selector continued from Qwen3-8B, MK2 at 32K, 200 prompts on an H100. Heads are added in the order of the MK3 calibration probe's score, labeled with the threshold that returns them, or drawn at random. The lines mark the shipped selector and full attention in every head, and the open marker is full attention in every head but the 11 flagged ones.

D PRECISION IS CHEAP, RESOLUTION IS NOT

A selector ranks blocks from a compact summary of the keys, which can give up resolution as pooled keys do, channels as low-rank projections do, or precision as quantized keys do. We use the notation of Section 3. A head's log mass of a block is $f_b - \log Z$ with the block log-sum-exp $f_b = \log \sum_{j \in \mathcal{B}_b} e^{s_j}$, so it ranks blocks by f_b. Exact mass selection keeps the K free blocks of largest group mass M_b in Equation 2. Each proposition is followed by its proof.

Proposition 1 (Pooling has a blind window)**.** *Let block b have logits s_0 on all keys except one, the needle, with logit $s_0 + \Delta$, and let block c have logit $s_0 + \delta$ on every key, with $\Delta, \delta > 0$. Exact mass ranks b above c if and only if $\Delta > \log\big(1 + L(e^{\delta} - 1)\big)$, which holds whenever $\Delta > \delta + \log L$.*

Any score that is an increasing function of the mean logit over a window of w keys ranks c above b whenever $\Delta < w\delta$. Every needle with $\delta + \log L < \Delta < w\,\delta$ is therefore ranked correctly by exact mass and incorrectly by the pooled score.

Proof. (i) By construction, $f_b = s_0 + \log(L - 1 + e^{\Delta})$ and $f_c = s_0 + \delta + \log L$, so $f_b > f_c$ if and only if $e^{\Delta} > 1 + L(e^{\delta} - 1)$. Since $Le^{\delta} - \big(1 + L(e^{\delta} - 1)\big) = L - 1 \geq 0$, $\Delta > \delta + \log L$ suffices. *(ii)* A score is linear in the key, so for the mean $\bar{\mathbf{k}}$ of a window W, $\mathbf{q}^{\top}\bar{\mathbf{k}}/\sqrt{d} = \frac{1}{w}\sum_{j \in W} s_j$ exactly. Pooling keys is averaging logits, which no nonlinearity downstream can undo. A window of b that holds the needle scores $s_0 + \Delta/w$ and every window of c scores $s_0 + \delta$. *(iii)* Any block score increasing in its window scores, which includes the sum, the maximum and InfLLM-V2's softmax over windows, ranks c above b whenever $\Delta/w < \delta$. With overlapping windows, every window holding the needle still scores $s_0 + \Delta/w$. □

The group. InfLLM-V2 scores a window by summing the scores of the query heads that share a key-value head, and exact mass sums their masses. When the other heads of the group are flat over both blocks, both sums rank b and c as the head that reads the needle does, so the proposition holds for the group.

Competitors raised on a few keys. If c is instead raised by δ on k of its L keys, the same argument gives exact mass ranking b first whenever $\Delta > \delta + \log k$, since $e^{\Delta} > 1 + k(e^{\delta} - 1)$ then holds. It gives a pooled score of the proposition's kind ranking c first whenever $\Delta < k_w\delta$, where $k_w \leq \min(k, w)$ is the largest number of raised keys that one window holds. This holds because c's best window then outscores the window that holds the needle and no window of c falls below s_0. The blind window $\delta + \log k < \Delta < k_w\delta$ is open once $(k_w - 1)\,\delta > \log k$, and $k = L$ recovers the proposition. In the flagged heads, the attention inside the needle block sits on about one key, an inverse participation ratio of 0.70 to 0.84, and inside the blocks at the exact selection cutoff it sits on two to three, 0.31 to 0.51 (Table 12). Concentration is not the mean logit that a pooled score reads, so we also measure the window itself.

The window on the data. Table 10 measures the proposition's quantities on the probed tokens of the calibration draw, in the query head of each group that puts the most attention on the needle block, with every window measured against the median window of that head over the prompt. The needle key stands 7 to 28 nats above the other keys of its block in every head we follow, and the free block at the pooled cutoff stands 1.3 to 5.6 nats above the median in every head. Look-alike blocks therefore compete everywhere, and what separates the heads is the rest of the needle's block. In the heads whose selector finds the needle, its other keys sit 3.9 to 11.7 nats above the median and the needle's line is raised as a whole. Its best window clears the cutoff, and the pooled score loses the needle block on 1 to 7 percent of the tokens. In the flagged heads of layers 9 and 16, those keys sit at or below the median, the proposition's case of one raised key in a flat block. The needle's best window rises only 0.9 and 1.4 nats against cutoffs of 1.3 and 2.2. The pooled score loses the needle block on 65 percent of the tokens while the head's exact block mass keeps it on 86 to 89 percent. The flagged head of layer 22 raises the needle's block as the heads whose selector finds the needle do and loses it on a quarter of the tokens, so the proposition accounts for two of the three flagged heads. These verdicts are one query head's, while the shipped selector sums the group and misses the needle block on 238, 227 and 72 of the 296 probed tokens in the flagged heads and on 7 to 52 in the others.

The true log mass follows the needle's logit with slope near one, and a pooled score follows it with slope $1/w$. InfLLM-V2's pooled keys are plain means of the same rotated keys that attention reads, with no learned projection, scored by the same rotated queries, so step (ii) holds for it exactly. Its first stage is a softmax over windows of these pooled keys, and its later stages are monotone in the window scores. The proposition therefore applies to it (Figure 7a).

Corollary 1 (the window widens with the context). Let the query token have N free candidate blocks with independent elevations $\delta_1, \ldots, \delta_N$, and let $\delta_{(K)}$ be the K-th largest, for K free slots. Exact selection keeps b when $\Delta > \log L + \delta_{(K)}$ and a pooled score keeps it only when $\Delta > w\,\delta_{(K)}$. The blind window therefore has width $(w - 1)\,\delta_{(K)} - \log L$, which grows with N at a fixed budget. For Gaussian elevations, $\delta_{(K)}$ sits near the $1 - K/N$ quantile, about 1.1σ for the 464 free candidates of an average 32K prompt and 1.5σ for the 979 of a 64K prompt with $K = 62$. Consistent with it, though any noisy top-K loses more as the candidates grow, the shipped selector's deficit on MK3 grows with the length inside each model's native range (Figure 5). The deficit grows from 3.0 points

Table 10: **Proposition 1 on the probed tokens** (InfLLM-V2-8B, 32K, the 40 calibration prompts of Figure 7b, 296 probed tokens per head). Per key-value head, in its query head with the most needle mass, the shipped selector's misses of the needle block and, over all probed tokens, the median gap Δ of the needle key over the other keys of its block, and the medians of those other keys (Needle Block), of the best window over the needle block (Needle Window) and of the window at that head's pooled cutoff among the other free blocks (Cutoff Window), all in nats over the head's median window on the prompt. The last columns give the fractions of tokens on which that head's exact block mass keeps the needle block, its pooled score loses it, and both. Windows are InfLLM-V2's, 32 keys at a stride of 16, five per block.

Head	Flagged	Shipped Misses	Δ	Needle Block	Needle Window	Cutoff Window	Exact Keeps	Pooled Loses	Both
L9.h0	yes	238 of 296	21.4	−0.6	0.9	1.3	0.89	0.65	0.56
L16.h0	yes	227 of 296	21.6	−0.1	1.4	2.2	0.86	0.65	0.53
L22.h0	yes	72 of 296	28.2	3.8	5.6	3.6	1.00	0.25	0.25
L23.h1	no	52 of 296	7.4	4.4	7.4	3.0	0.94	0.07	0.03
L26.h0	no	7 of 296	18.4	11.7	15.5	5.6	1.00	0.02	0.02
L26.h1	no	7 of 296	9.5	9.0	12.2	4.4	1.00	0.01	0.01
L29.h1	no	9 of 296	14.8	3.9	5.9	2.5	1.00	0.01	0.01

at 16K through 6.0, 18.0 and 22.0 to 41.5 at 64K on MiniCPM4.1-8B, and from 10.5 at 16K through 22.5 to 33.0 at 32K on InfLLM-V2-8B. By contrast, full attention in the flagged heads and the repair at every token stay within a point of full attention at every length within each model's native range. On real API data, the deficit grows from none below 16K to −8.6 at a median of 56K (Appendix H). On InfLLM-V2-8B at 64K, beyond its native range, full attention falls too. Further out, on 100 prompts of one draw, full attention in the blind heads keeps 46.0 of the 68.0 that full attention reads on InfLLM-V2-8B at 96K, against 4.0 under the shipped selector. At 128K, four times its native range, the model answers almost nothing under any arm. MiniCPM4.1-8B at 96K reads 21.0 under the shipped selector and 82.0 with full attention in the blind heads, level with full attention at 81.0. At 128K, twice its native range, it reads 10.0, 45.0 and 51.0, where the paired test separates the blind heads' gain from zero and not their gap to full attention.

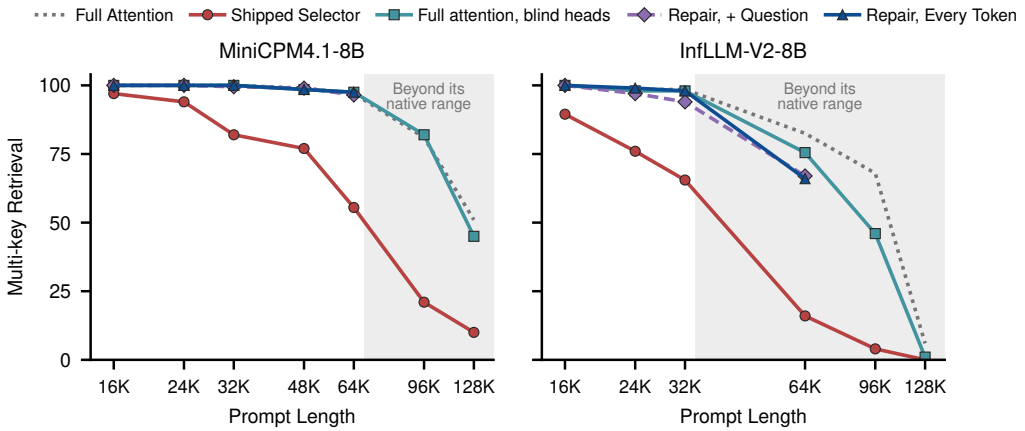


Figure 5: **The deficit grows with the length** (MK3 at the shipped budget, 200 prompts per point, 100 at 96K and 128K). The 16K, 24K and 48K points are draws of the same generator seed as the 32K and 64K cells of Table 2, and the flagged heads are those calibrated at 32K. The shaded ranges lie beyond each model's native range.

Where in the prompt the needle is lost. Figure 6 splits every cell of the length sweep by the needle's depth in the prompt. Within each model's native range, the shipped selector loses needles at every depth, most in the second fifth of the prompt at 32K on InfLLM-V2-8B, and full attention in the blind heads returns every band to within a few points of full attention. Beyond the native range, the early bands go first. At 64K on InfLLM-V2-8B, the shipped selector loses almost every needle in the first two fifths. Full attention in the blind heads keeps most of them, and its remaining gap to full attention sits in those same bands. At 96K, the same order holds at lower values. On MiniCPM4.1-8B at 128K, the first fifth is lost under every arm, and in the other bands full attention in the blind heads reads about what full attention does.

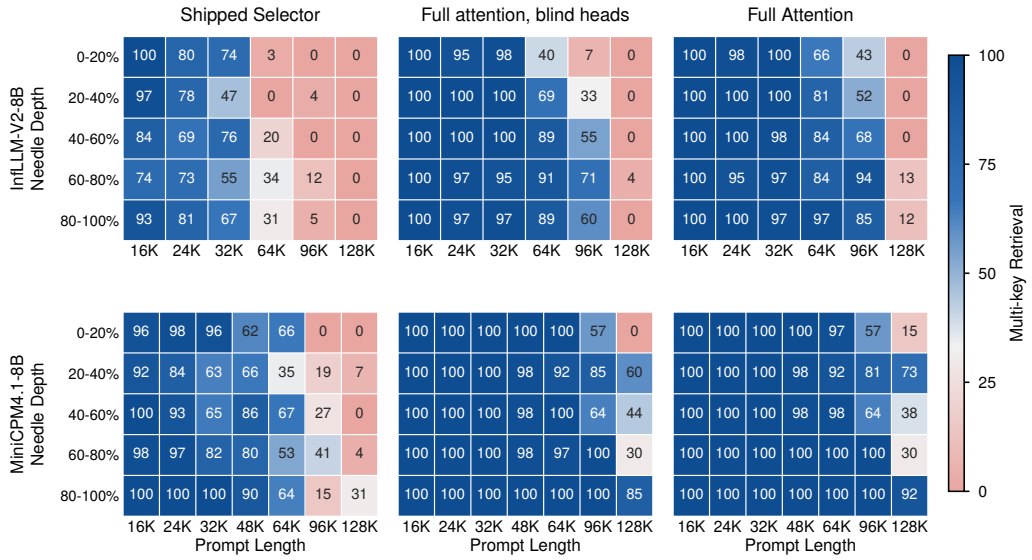


Figure 6: **The loss by needle depth.** MK3 accuracy in five bands of the needle's position in the prompt at each length of Figure 5, for the shipped selector, full attention in the blind heads and full attention, about 40 prompts per cell within the native ranges and 20 at 96K and 128K.

Corollary 2 (the repair is a small edit). Under the model of Proposition 1, the pooled and the exact selection of a token differ only on blocks that hold a needle, so if a token has r such blocks outside the forced set, replacing its r weakest free picks by the r heaviest blocks its selection missed recovers exact selection. On InfLLM-V2-8B, replacing the single weakest pick of each prompt token, 0.81 blocks for an average token, recovers 31.5 of the 32.5 points that replacing all 62 free picks recovers (Figure 7c).

Proposition 2 (Quantization moves a block score by at most the largest logit error)**.** *Let* $\hat{\mathbf{k}}_j = \mathbf{k}_j + \boldsymbol{\varepsilon}_j$ *be the keys used for scoring, with logit errors* $e_j = \mathbf{q}^\top \boldsymbol{\varepsilon}_j / \sqrt{d}$*. Then* $|\hat{f}_b - f_b| \le \max_{j \in \mathcal{B}_b} |e_j|$ *for every block, and the order of two blocks is preserved whenever their margin exceeds* $2 \max_j |e_j|$*. If the errors are uncorrelated across channels with* $\mathbb{E}\,\varepsilon_{jc}^2 = D\,\sigma_c^2$*, where* σ_c^2 *is the variance of channel* c *over the keys, then* $\mathbb{E}\,e_j^2 = D\,\sigma_s^2$ *with* $\sigma_s^2 = \frac{1}{d}\sum_c q_c^2 \sigma_c^2$*, which is the variance of the exact logits over keys when the key channels are uncorrelated.*

Proof. The gradient of $\log \sum_j e^{s_j}$ is the softmax of s, a probability vector, so the map is 1-Lipschitz from the supremum norm and $|\hat{f}_b - f_b| \le \max_{j \in \mathcal{B}_b} |e_j|$. Two block scores that differ by more than $2 \max_j |e_j|$ each move by less than half the gap. With $e_j = \sum_c q_c \varepsilon_{jc} / \sqrt{d}$ and uncorrelated channel errors, $\mathbb{E}\,e_j^2 = \frac{1}{d}\sum_c q_c^2 D \sigma_c^2 = D\sigma_s^2$, the variance of the exact logits over keys times D when the channels are uncorrelated, independent of the query, the head and the length. □

The average-case part assumes uncorrelated errors of the stated size, which clipped entries break, 2.6 percent of the flagged heads' key entries against 0.7 percent for a Gaussian source (Appendix E). In contrast, the worst-case bound holds for any error.

For a single head, every block shares one normalizer, so the ranking by mass is the ranking by f_b. The group sum of Equation 5 adds heads whose normalizers move by different amounts, each by at most the largest logit error over all of that head's keys. A block's group score therefore moves by at most twice that error, and its ranking holds above a margin, in log mass, of four times it.

The bound holds for any summary that keeps one entry per key. Keeping only the channels in a set $\mathcal{C}$ leaves logit error of variance $(1 - \rho_{\mathcal{C}})\sigma_s^2$, with $\rho_{\mathcal{C}}$ the share of the logit variance they carry (Yang et al., 2024), while quantizing leaves a fixed fraction $\sqrt{D}$ of the logit spread. For the per-channel uniform quantizer of Equation 4, D is the distortion of the quantizer on the standardized keys, which the Gaussian step fixes at each bit width (Max, 1960). Every additional bit roughly halves the noise (Table 11). If γ is the margin of the needle block over the last selected competitor, the needle is

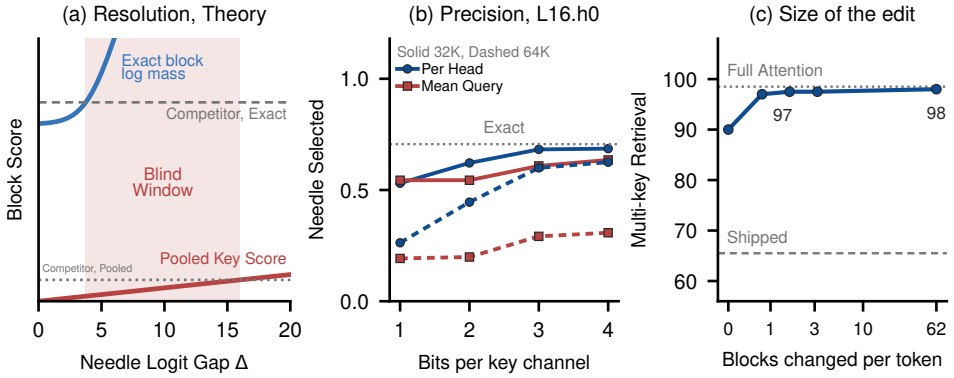


Figure 7: **Resolution, precision, and the size of the edit they imply** (InfLLM-V2-8B). **(a)** Proposition 1 drawn for L=64 and w=32 in its idealized case. A competitor better by δ=0.5 nats on every key hides every needle between the two crossings, from 3.8 to 16 nats. Table 10 measures the window on the probed tokens. **(b)** Needle recall against bit width for the per-head criterion and for the softmax of the group-mean query, at both lengths, 296 probed tokens per head at 32K and 312 at 64K, those of 40 calibration prompts whose needle block is not forced. **(c)** Multi-key retrieval against blocks changed per prompt token, n=200 prompts. One block carries almost all of it.

Table 11: **Proposition 2 against the probe** (InfLLM-V2-8B, 32K, the five key-value heads of Table 12). Predicted noise is $\sqrt{D}$ of the optimal uniform quantizer of a unit Gaussian (Max, 1960), computed numerically at the steps the repair uses. Measured noise is the standard deviation of the logit error of the quantized keys over that of the exact logits, and recall is the needle recall of the per-head criterion on the quantized keys minus that of exact mass, both as ranges over the five heads. The last column is the mean absolute error of the Gaussian margin model's predicted recall over heads and both query summaries, 0.04 over all 40 combinations.

Bits per Channel	Predicted Noise	Measured Noise	Recall against Exact Mass	Model Error
1	0.60	0.55 to 0.64	−0.24 to −0.03	0.10
2	0.34	0.32 to 0.37	−0.07 to +0.00	0.03
3	0.19	0.18 to 0.21	−0.02 to +0.00	0.02
4	0.11	0.10 to 0.12	−0.01 to +0.00	0.01

dropped with probability about $\Phi(-\gamma/\sigma_e)$ with $\sigma_e = \sqrt{D}\,\sigma_s$, which has a sharp knee in the number of bits.

Table 12: **The five probed key-value heads** (InfLLM-V2-8B, MK3, 32K). Recall of the needle block under the shipped selector and the group's mean needle mass, from the calibration probe. Query heads on the needle counts, per token, the query heads of the group with more than five percent of their mass on the needle block, and concentration is the inverse participation ratio of the attention inside the needle block and inside the eight blocks around the cutoff of exact mass, 1 when one key holds a block's mass, all averaged over the query tokens of the theory probe. The last two columns give, for the three query heads with the most needle mass, how often the needle block is among their own top free blocks at token level and under the shipped pooled keys, over the 216 probed tokens of 30 calibration prompts whose needle block is not forced.

					Concentration		Selected	
Head	Flagged	Sparse Recall	Needle Mass	Query Heads on the Needle	Needle Block	Cutoff Blocks	Token Level	Pooled
L9.h0	yes	0.19	0.080	2.2	0.81	0.51	0.74	0.18
L16.h0	yes	0.24	0.056	2.0	0.70	0.31	0.71	0.25
L22.h0	yes	0.73	0.192	5.4	0.84	0.37	0.97	0.75
L23.h1	no	0.83	0.069	3.8	0.18	0.14	0.85	0.85
L26.h0	no	0.98	0.287	9.2	0.72	0.37	1.00	0.99

Predictions and measurements. Every probe measurement in this appendix reads the calibration draw of seed 7, which shares no prompt with the evaluation draws. The bit width therefore does not rest on the reported prompts, although the sweep of Table 13 on the first evaluation draw was seen before it was fixed. For five probed key-value heads, the measured logit noise matches the predicted fraction of the logit spread at every bit width, and the per-head criterion keeps the needle as well as

exact mass does from three bits on in every head. A Gaussian margin model driven by the measured margins and noise predicts the recall of every combination of head, query summary and bit width closely (Table 11). At 64K, with twice as many competitor blocks, the knee moves to three or four bits (Figure 7b), which is why the repair uses four. At that length, the softmax of the group-mean query loses a needle that the per-head criterion keeps. In the flagged heads of layers 9 and 16, the shipped selector places a mean of 216 and 190 free blocks ahead of the needle block, of about 460 candidates, against 46 and 61 under exact mass. The pooled score is therefore close to blind to the needle rather than just short of the cutoff, which is why more budget recovers the loss only gradually (Appendix E). End to end in the flagged heads, the criterion stays within the noise of exact mass down to two bits per channel and falls below the shipped selector on keys mean-pooled over a block (Table 13), so a selector's key summary can spare precision but not resolution. Holding the per-head criterion fixed in the flagged heads at the decode tokens and averaging the keys over windows of 4, 8, 16 and 64 tokens lowers accuracy monotonically from 89.5 with single keys to 86.0, 84.0, 71.5 and 59.5. For comparison, the shipped selector, whose windows of 32 at a stride of 16 fall between the last two, scores 65.5, and the criterion with 4-bit keys scores 90.0. Adding look-alike competitors at a fixed length does what the proposition predicts. The shipped selector scores 100.0 with one needle in noise, 93.0 with three look-alike needles in an essay, and 85.5 and 65.5 when every line of the haystack is a look-alike pair of words and numbers or of UUIDs, against 98.0 to 100.0 with full attention (Table 16).

E Ablations, per-task results and comparisons

All cells in this section are InfLLM-V2-8B on MK3 at 32K with $n{=}200$ unless stated, where the shipped selector scores 65.5 and full attention scores 98.5.

Every design choice. Table 13 varies one choice of the every-token repair at a time and adds the negative controls. Table 14 sweeps the tokens it acts at. Restricted to a single flagged head, the question-token repair scores 80.0, 86.0 and 82.0 in the heads of layers 9, 16 and 22 against 94.0 for the three together, so the heads are complementary. The repair in every head at the decode and question tokens scores 96.0, below the three heads at every token, so where the scorer acts matters more than in how many heads. The per-head criterion and the softmax of the group-mean query are level at 32K. At 64K, the mean query loses needles that the per-head criterion keeps (Figure 7b). At 32K, the default sits within a point of full attention, so these end-to-end ablations separate only choices that cost a few points. The probe of Figure 7b measures the bit width and the criterion away from that ceiling, as needle recall on single tokens.

Form of the quantizer. The copy quantizes the cached keys, after the rotary embedding. A uniform code of 2^b levels trades rounding error inside its range against clipping outside it. The Gaussian step minimizes the two together for a Gaussian of unit variance, the model that assumes nothing about a channel beyond its mean and standard deviation. The group of Table 13 on the quantizer runs on an H100 against the default rerun there, 96.5. At four bits, the grid barely matters. Halving or doubling the Gaussian step, which moves the clamp to 1.34 or 5.36 standard deviations, scores 95.0 and 98.0. Each channel's own range as KIVI sets it (Liu et al., 2024b) scores 98.0, and one mean and standard deviation for the whole head scores 96.5. At two bits, the grid matters, and the Gaussian step scores 95.0 against 91.5 for the min-max grid. The flagged heads' keys are not Gaussian. Standardized by the statistics of the first 6144 keys, 2.6 percent of their entries fall outside the 4-bit range against 0.7 percent for a Gaussian, and their median channel has an excess kurtosis of -0.61. The step that minimizes their squared error is at least twice Max's, whose error is 2.1 times that minimum. The ranking tolerates this because a block score moves by at most its largest logit error (Proposition 2). At four bits, the criterion keeps the needle as well as exact mass does (Table 11). At two bits, where the grid matters, the step that minimizes their squared error is 1.28 times Max's, and Max's error is within 6 percent of that minimum.

Budget. More budget reaches the repair only at more than twice the blocks per token. With 104, 112, 128, 160, 192, and 224 blocks per token instead of 96, at prefill and at decode, the shipped selector scores 70.0, 75.5, 80.5, 91.0, 94.5, and 95.5. Moving the decode budget between layers at matched total reads scores 71.5 to 77.5. Giving only the three flagged layers two or four times their free blocks at the decode tokens, filled from the shipped ranking, scores 78.0 and 88.0, below the 90.0 of the repair at the decode tokens at the shipped budget (Table 2), so in these heads, what is

Table 13: **Ablations and negative controls of the every-token repair** (InfLLM-V2-8B, MK3, 32K, n=200). Each group varies one choice and holds the others at the default, which is set in bold. Differences are paired over the same 200 prompts, and the groups run on an H100 are compared with the default and the shipped selector rerun there. A dash in the middle column is the default compared with itself. Bold marks the default here, not the best.

Setting	Accuracy	Against the Default	Against Sparse
Heads that carry the scorer (threshold of the flagging rule)			
2 heads (τ=0.03)	95.0	−3.0	+29.5
3 heads (τ=0.01, the default)	98.0	–	+32.5
10 heads (τ=0.001)	99.0	+1.0	+33.5
All 64 heads	99.5	+1.5	+34.0
Form of the flagging rule (its three highest-scoring heads)			
Recall gap × needle mass (default), layers 9, 16, 22	98.0	–	+32.5
Recall gap alone, L6.h0, L6.h1, L9.h0	82.0	−16.0	+16.5
Needle mass alone, L22.h0, L26.h0, L26.h1	80.5	−17.5	+15.0
Retrieval score of Wu et al. (2024b), L22.h0, L26.h0, L29.h1	81.0	−17.0	+15.5
Precision of the scanned keys			
1 bit	87.0	−11.0	+21.5
2 bits	97.0	−1.0	+31.5
3 bits	98.5	+0.5	+33.0
4 bits (default)	98.0	–	+32.5
16 bits (exact mass)	98.5	+0.5	+33.0
Criterion			
Sum of the per-head softmaxes (default)	98.0	–	+32.5
Softmax of the mean query	97.0	−1.0	+31.5
Keys that set the quantizer's statistics			
The first 512	98.5	+0.5	+33.0
The first 6144 (default)	98.0	–	+32.5
The whole prompt (not causal)	97.0	−1.0	+31.5
Form of the quantizer (H100, against the default rerun on H100)			
Gaussian step, per channel, 4 bits (default)	96.5	–	+29.5
Step halved, clamp at 1.34 standard deviations	95.0	−1.5	+28.0
Step doubled, clamp at 5.36 standard deviations	98.0	+1.5	+31.0
Each channel's own range, the min-max grid of KIVI	98.0	+1.5	+31.0
One mean and standard deviation for the whole head	96.5	+0.0	+29.5
2 bits, Gaussian step	95.0	−1.5	+28.0
2 bits, each channel's own range	91.5	−5.0	+24.5
Negative controls			
The scorer in the *other* head of layers 9, 16, and 22	64.5	−33.5	−1.0
Mean-pooled keys, one per 64 tokens, in the three heads	53.5	−44.5	−12.0
Random free blocks in the three heads, decode tokens	12.5	−85.5	−53.0
Random free blocks in the other heads of those layers, decode tokens	64.5	−33.5	−1.0
Negative controls on H100, against the default and the shipped selector rerun there			
The scorer in the 61 *unflagged* heads	66.5	−30.0	−0.5
Full attention in three random unflagged heads, draw 0 (L12.h1, L25.h1, L28.h1)	66.0	−30.5	−1.0
Full attention in three random unflagged heads, draw 1 (L4.h0, L19.h0, L28.h1)	63.5	−33.0	−3.5
Full attention in three random unflagged heads, draw 2 (L28.h1, L29.h0, L31.h1)	66.5	−30.0	−0.5

missing is selection, not budget. Cutting the free slots of every decode token in every head from 62 to 46 and to 30 drops the shipped selector to 58.0 and 44.0, while the every-token repair holds at 98.0 and 98.0. On this one draw of MK3, a decode selection of 64 blocks does for the repair what the shipped selection of 96 does.

Which tokens. Once calibration fixes the heads and the theory fixes the bit width, the tokens the repair acts at remain. Table 14 sweeps them on a task the repair helps and one it hurts. Rewriting more of the prompt buys multi-key retrieval, from $+24.5$ at the decode tokens alone to $+32.5$ at every token. Common word extraction gains a little while the scorer rewrites up to 512 prompt tokens. From 2048 on, where the question ends and the context begins, it loses 0.4 to 5.5 points. The loss does not come from discarding the shipped selection's coverage, because changing a single block in each token costs at least as much as replacing all 62 picks. Exact mass in place of the 4-bit scorer costs the same again. On HELMET's three retrieval-augmented question answering sets (Yen et al., 2024) rerun at n=600, where the shipped selector has no consistent deficit, the every-token repair costs -2.8 on PopQA and -1.5 on Natural Questions and gains $+1.2$ on HotpotQA. Repairing the decode tokens alone costs -0.3.

Where full attention is not the ceiling. On MiniCPM4.1-8B variable tracking, the dense arm collapses at the top of the model's native range and the shipped selector does not. At 32K, the two

Table 14: **The repair's one knob** (InfLLM-V2-8B, n=200, three flagged heads, 4-bit keys unless noted). Prompt tokens rewritten, against multi-key retrieval which the repair helps, and common word extraction which it can hurt. Differences are paired against the shipped selector on the same draw, which scores 65.5 and 45.9. The exact mass row is excluded from the ranking as well.

	Prompt Tokens	MK3		Common Words	
Tokens the Scorer Acts at	Rewritten	Score	vs Sparse	Score	vs Sparse
Decode tokens only	0	90.0	+24.5	46.40	+0.5
+ the last 128, the question	128	94.0	+28.5	**48.25**	+2.4
+ the last 512	512	94.0	+28.5	<u>48.20</u>	+2.3
+ the last 2048	2048	95.5	+30.0	43.35	−2.5
+ the last 8192	8192	94.5	+29.0	45.50	−0.4
Every token, one block changed	all	97.0	+31.5	41.45	−4.4
Every token, two blocks	all	<u>97.5</u>	+32.0	40.35	−5.5
Every token, all 62 free picks	all	**98.0**	+32.5	42.70	−3.2
Every token, all 62, exact mass	all	98.5	+33.0	41.45	−4.4

Table 15: **Block scores that do not help** (InfLLM-V2-8B, MK3, 32K, n=200, the free slots of the decode tokens of every head). The shipped first stage re-implemented on the shipped pooled keys with three ways of aggregating the group, and Quest's per-channel bound. Quest's bound covers a block's largest logit and so does not dilute a needle. Yet at 64-token blocks, it bounds the competitors loosely and falls below the shipped selector, a failure of looseness rather than of resolution. The exact mass reference is excluded from the ranking as well.

Score in the Free Slots	Accuracy
Sparse, as released	<u>65.5</u>
Sum over the group's query heads, the sparse rule	65.0
Maximum over the group's query heads	<u>65.5</u>
Sum over the two query heads with the largest scores	**67.0**
Quest's per-channel bound, mean query	33.0
Quest's per-channel bound, maximum over the query heads	29.0
Exact mass	90.0

are level. At 64K, they are 46.9 against 74.4, with the dense arm below on 92 of 200 prompts, answering with invented names in place of the tracked variables. Fu et al. (2026) report the same inversion, 54.8 against 74.4, and our shipped arm reproduces theirs to the decimal on this task and on multi-value retrieval. This is why the 12-task mean of MiniCPM4.1-8B at 64K closes while the eleven-task mean does not, and why we do not describe full attention as a ceiling on that mean. Full attention in the flagged heads inherits part of the collapse, 56.8 on variable tracking over 50 prompts, which puts its 12-task mean in Table 3 below the repair's. Over all 200 prompts of each task (Table 16), the question-token variant reaches a 12-task mean of 86.8 on the same checkpoint, against 87.3 for the every-token variant.

The attention sink (Xiao et al., 2023) on SeerAttention. SeerAttention's gate forces no block, and the per-head criterion on 4-bit keys with nothing forced scored 0.0 on the first 25 prompts, because the first token's key is an outlier that the 4-bit code clips. Keeping the first block and the query's own block, as the InfLLM-V2 kernel does, a change made after that run on evaluation prompts, gives 92.5, what exact mass scores on the same GPU type in Table 5.

Our harness against SparDA's. On the 12 RULER tasks that both evaluations run for MiniCPM4.1-8B at 64K, our full attention and shipped cells differ from those Fu et al. (2026) report by at most 8.5 and 6.5 points on any task, on question answering and on MK3, differences of the size that their 50 prompts per task leave open. The 12-task means are 86.9 and 80.2 against their 88.2 and 81.0. Their NOSA-8B rows use 96 blocks per token where the release and our runs use 64, so their shipped cells are not a reproduction of ours. Full attention does not depend on the budget, and our 12-task mean there is 89.1 against their 89.6.

Table 16: **RULER per task at the shipped budget** (n=200 per task, 32K for InfLLM-V2-8B and 64K for MiniCPM4.1-8B). "Dense, Flagged" gives the three flagged heads full attention, and "Exact" is exact mass selection at the decode tokens. "+ Question" is the repair at the decode tokens and the last 128 prompt tokens, and "Every Token" is the default repair. The frequent-words prompts average 60,851 tokens rather than 65,536, because RULER's generator undershoots the target with this tokenizer, identically for every arm.

	InfLLM-V2-8B, 32K						MiniCPM4.1-8B, 64K			
Task	Dense	Sparse	Dense, Flagged	Exact	+ Question	Every Token	Dense	Sparse	+ Question	Every Token
Single 1 to 3	100.0	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	100.0	**100.0**	**100.0**	**100.0**
MK1	99.0	93.0	**99.0**	94.0	97.5	**99.0**	98.5	91.0	98.0	**99.5**
MK2	98.0	85.5	**98.5**	90.5	97.5	**98.5**	99.5	80.5	98.5	**100.0**
MK3	98.5	65.5	**98.0**	90.0	94.0	**98.0**	97.0	55.5	96.5	**97.5**
Multi-value	98.9	98.8	**99.2**	97.1	98.2	**99.2**	97.9	95.5	98.5	**98.8**
Multi-query	97.5	96.2	96.4	**97.1**	96.8	96.5	99.6	95.8	99.0	**99.5**
Variable Tracking	99.0	**99.7**	**99.7**	**99.7**	**99.7**	**99.7**	46.9	74.4	72.7	**74.5**
Common Words	53.2	45.9	34.0	**50.0**	48.2	42.7	40.5	11.8	14.5	**14.7**
Frequent Words	94.2	87.8	**89.2**	86.2	88.0	87.7	91.3	89.5	**91.2**	88.7
QA 1	70.5	71.5	71.0	**72.5**	70.5	69.5	71.5	68.0	73.0	**75.0**
Mean	92.4	87.0	90.4	89.8	**90.9**	**90.9**	86.9	80.2	86.8	**87.3**

Baselines in our harness. The rows of Table 3 between Sparse and the repair run in our harness at the shipped budget, in every head, through the same hook as the repair. Quest (Tang et al., 2024b) scores a block by the bound that the per-channel extremes of its keys give the group-mean query. Double Sparsity (Yang et al., 2024) scores it by the group's attention on 16 label channels, and Loki (Singhania et al., 2024) scores it on 32 principal directions, both chosen from each prompt's own keys, which favors them over the offline calibration the methods ship with. All three score the free slots at the decode tokens and the last 128 prompt tokens, the token set of the repair's cheaper variant. SnapKV (Li et al., 2024) fills the free slots of every decode token from the blocks that its 32-token observation window at the end of the prompt attends to most. "Dense, flagged heads" gives the flagged heads full attention at every token, in the prefill and at decode, as DuoAttention does for its retrieval heads (Xiao et al., 2024). Its cost is in Appendix F. Quest runs at the model's 64-token blocks with a group query, as the kernel requires, far from its per-head 16-token pages. Our rows use 200 prompts per task, 100 for the NOSA-8B tasks other than MK3 and 50 for the baselines' tasks other than MK3. The RULER means of Table 3 are paired over the first 50 prompts of each task besides MK3, which every row shares. Fu et al. (2026), whose SparDA trains a 33.5M-parameter selector, report from 50 prompts per task in their own harness, with InfiniGen (Lee et al., 2024) adapted to block selection and NOSA-8B at 96 blocks per token where the release and our runs use 64. They report MK3 scores and 12-task means of 28.0 and 71.0 for InfiniGen and 56.0 and 81.6 for SparDA on MiniCPM4.1-8B at 64K, and of 8.0 and 67.4 and 28.0 and 76.6 on NOSA-8B at 32K. Every one of these eight numbers is below the repair at every token and full attention in the flagged heads (Table 3). The bytes per cached token of Table 3 follow from each method's definition with 64 key-value heads of dimension 128 in bfloat16.

A trained indexer. Before the head-level diagnosis, we trained a lightweight indexer as a baseline, projections of rank 32 per layer and key-value head and a shared two-layer scorer, 0.53M parameters in the lightning-indexer family of DSA and SpotAttention (DeepSeek-AI, 2025; Ahmad & Yun, 2026). It is trained by key-level cross-entropy against the shipped model's own attention on unlabeled LongBench v1 contexts (Bai et al., 2023), 82 minutes on one A100. Its top blocks fill every free slot at the shipped budget. On MK3, it scores 70.5 on InfLLM-V2-8B at 32K, and 71.0 and 76.0 with two more training seeds, against 65.5 shipped and 98.0 repaired. On the same task, it scores 13.0 on the retraining against 23.5 shipped and 84.0 repaired, and it scores 89.5 on Dash-8B at 16K against 64.5 shipped and 94.5 with the scorer in every head. On SeerAttention, it scores 35.5 against 71.5 shipped and 92.5 repaired, and 84.5 once the attention sink and the query's own block are forced into its picks.

F MEASURED COST

Memory. With a head dimension of 128 and 4 bits per channel, the side cache of three flagged heads costs 192 bytes per token against 32,768 bytes of KV cache, which is 0.6 percent.

Arithmetic. The every-token repair scores one product with every earlier key for each of the 16 query heads of a flagged key-value head and each token past position 6144, where its statistics are fixed. At 32K with three heads, that is 3.2×10^{12} multiply-accumulates, 1.1 percent of the prefill of the 8B model. At 64K with four heads, it is 2.9 percent. Tokens below position 8192 read every block on these checkpoints, so this counts some work that changes no selection. Unfused, element-wise passes over the score matrices cost 17.4 percent to the first token on an A100 and 25.4 on an H200. Fusing the masked block reduction brings this to 5.2 and 6.9 percent (Table 17), while the fused path scores 99.0 against 98.0 for the emulation. In absolute terms, the fused repair adds 2.1 to 3.2 ms per output token on the three GPU types, against a shipped time of 46 to 68 ms per token that the framework's overhead dominates. In a faster serving engine, the same absolute cost would therefore be a larger share.

Full attention in the flagged heads. At batch size one, full attention in the three flagged heads costs less than the repair. Beside it in the same run on an H100, it changes the time per output token by -1.0 percent at 32K and $+0.4$ percent at 64K against $+4.4$ and $+4.7$ percent for the repair. It changes the time to the first token by $+1.9$ and $+2.9$ percent against $+6.6$ and $+9.5$ percent. It needs no side cache, because it reads the cache the model already keeps. What it reads per decoded token is the 16-bit keys and values of the flagged heads at every position, eight times the bytes of the 4-bit keys that the repair scans. At batch size one, the model's weights dominate the bytes read per token.

Deployment. Full attention in the flagged heads needs only a per-head exception in the block list, since those heads read every block of the cache the model already keeps. The scorer adds a side cache of 4-bit codes in the three or four layers that hold flagged heads, whose quantizer statistics are fixed per sequence once its first 6144 keys exist. It also adds a top-k over its block masses in place of the shipped selection in those heads. Every flagged head is the first key-value head of its layer. Under tensor parallelism over the two key-value heads, the extra work of either fix therefore lands on one rank, a load imbalance. We implement neither fix in a serving engine with paged caches, continuous batching, chunked prefill or prefix caching.

A hybrid of the two fixes. The prefill is bound by compute and decoding is bound by the bytes it reads, so the two fixes can split the work. The hybrid runs full attention in the flagged heads during the prefill and the scorer at the decode tokens, and keeps the shipped decode budget, as RTPurbo does for the retrieval heads of dense models with a trained indexer (Zhou et al., 2026). On MK3, the hybrid scores 97.5 on InfLLM-V2-8B at 32K, 96.5 on MiniCPM4.1-8B at 64K and 93.5 on NOSA-8B. Full attention in the flagged heads scores 98.0, 97.5 and 94.0 there, and the scorer at the decode tokens alone scores 90.0, 80.0 and 34.5. On NOSA-8B, whose loss is made across the prefill, the hybrid also closes the gap by which the repair at every token, 90.0, trails full attention. Timed at batch size one on an H100 beside its own shipped arm, it adds 1.8 and 2.6 percent to the time to the first token at 32K and 64K, as full attention in the flagged heads does. It adds 4.8 and 3.4 percent to the time per output token, as the scorer does (Table 17). Its reads per decoded token are those of the 4-bit scan (Table 18).

Many sequences at once. Table 18 times one decode step of one flagged head in isolation on an H100, attention kernels only on synthetic tensors. Each call is captured in a CUDA graph after a 256 MB write that flushes the L2 cache. Full attention in the head costs its increment over the sparse read of 64 selected blocks, as the retraining and NOSA-8B read, with FlashAttention 3 (Shah et al., 2024). At the 96 blocks of the other InfLLM-V2 checkpoints, the increment is smaller, by at most 7 percent at 32K. The repair costs a grouped Triton scan that reads each sequence's 4-bit codes once for the 16 query heads of the group, and then the group mass and the top-k over blocks. Its selections agree with a reference implementation. The scan alone is cheaper than the extra reads of full attention at every size. With its unfused group mass and top-k, the repair costs more at one to four sequences, and costs less from 16 sequences at 64K and 128K and from 64 at 32K. At 64 sequences of 128K, the three flagged heads of InfLLM-V2-8B add about 4.0 ms per step with full attention and 1.8 ms with the repair. Every accuracy cell comes from the harness's emulation, which scores per query head as the per-head path of Table 17 does. The fused path of that table scores 99.0

Table 17: **Measured cost** on InfLLM-V2-8B with 32K prompts, 64K where the label says so, greedy generation in HuggingFace transformers (Wolf et al., 2020) on one GPU at batch size one, six prompts after one warm-up, time per output token from a 64-token and a 1-token call. The deployable path packs the low-bit keys into a side cache, scans them with one Triton kernel per flagged head at decode, and scores the rewritten prefill tokens with matrix products on the dequantized keys. The next group runs the repair and full attention in the three flagged heads in one benchmark per length, and the last runs the hybrid of Appendix F in its own. Every arm is compared with the shipped arm of its own benchmark run.

		ms per Output Token			Time to First Token (s)		
GPU	Repair Arm	Sparse	Arm	Change	Sparse	Arm	Change
A100	Decode tokens, 3 heads, 4-bit keys, per query head	65.2	67.6	+3.6%	4.70	4.73	+0.6%
A100	Same, with the last 128 prefill tokens (the cheaper variant)	65.2	67.5	+3.5%	4.70	4.84	+2.8%
A100	Same, at every token, float16 scoring	66.4	69.6	+4.8%	4.72	5.54	+17.4%
A100	**Same, at every token, fused float16 scoring (the repair)**	67.9	71.1	+4.8%	4.73	4.97	+5.2%
H100	Decode tokens, 3 heads, 4-bit keys, per query head	47.6	49.8	+4.6%	2.00	2.00	−0.0%
H100	Same, with the last 128 prefill tokens (the cheaper variant)	47.6	49.6	+4.4%	2.00	2.08	+3.6%
H100	Same, at every token, float16 scoring	47.0	49.1	+4.5%	2.00	2.49	+24.4%
H100	**Same, at every token, fused float16 scoring (the repair)**	47.1	49.2	+4.5%	2.01	2.13	+6.4%
H200	Decode tokens, 3 heads, 4-bit keys, per query head	46.1	48.4	+5.0%	1.83	1.83	−0.1%
H200	Same, with the last 128 prefill tokens (the cheaper variant)	46.1	48.3	+4.7%	1.83	1.89	+3.4%
H200	Same, at every token, float16 scoring	46.1	48.2	+4.7%	1.83	2.30	+25.4%
H200	**Same, at every token, fused float16 scoring (the repair)**	46.4	48.7	+5.0%	1.83	1.96	+6.9%
H100	32K, the repair, fused, at every token	52.1	54.4	+4.4%	2.03	2.16	+6.6%
H100	32K, full attention in the three flagged heads	52.1	51.6	−1.0%	2.03	2.07	+1.9%
H100	64K, the repair, fused, at every token	57.3	60.0	+4.7%	4.47	4.89	+9.5%
H100	64K, full attention in the three flagged heads	57.3	57.6	+0.4%	4.47	4.59	+2.9%
H100	32K, the hybrid, full attention in the prefill, the scorer at decode	49.6	52.0	+4.8%	2.00	2.04	+1.8%
H100	64K, the hybrid, full attention in the prefill, the scorer at decode	54.7	56.5	+3.4%	4.42	4.53	+2.6%

on MK3 against the emulation's 98.0. The grouped kernel is timed here and checked against the reference but never runs end to end. At 32,768 bytes of cache per token, 16 sequences of 64K fit on one 80 GB GPU beside the weights, where the scorer saves about 0.1 ms per decode step over full attention's extra reads in three heads. The larger configurations need several GPUs' worth of cache. We do not time a batched decode end to end, but a byte count bounds the gain. A decode step of those 16 sequences reads about 20 GB with the weights, 6.6 ms at the rate this H100 reaches in a device copy. Full attention in the three flagged heads adds 1.5 GB to it against 0.2 GB for the scan. A repair whose group mass and top-k cost nothing would thus save at most 6 percent of the step, and the kernels above save 1.4 percent. At 64 sequences of 64K, the same count allows at most 13 percent, and the kernels save 7 percent. Where the scorer must act in every head, as on Dash-8B and NSA-8B, its scan of an eighth of every cached key's bytes outgrows the blocks a sparse head reads at long context, so the few-head case is where it pays.

G WHERE SELECTOR REPAIR STOPS

NOSA-8B. NOSA's attention adds a query-agnostic eviction score to every logit, and its kernel reads 18 forced blocks of the 64 it reads per token. Exact mass here is computed with that score and keeps those blocks. On MK3 at 32K, the shipped selector scores 18.5, exact mass at the decode tokens scores 34.5, and exact mass at every token scores 94.0. That matches full attention's 94.0, with 184 of 200 generations identical to it. The loss is thus made in the prefill and a selector can repair it. The probe flags five heads, the three of the InfLLM-V2 family and L26.h0 and L29.h1. In them, the repair scores 34.5, 79.5 and 90.0 at the decode tokens, with the question tokens and at every token. Exact mass in the same five heads also scores 90.0 at every token (Table 19). A keep-only check reproduced the release's selections.

InfLLM-V2-8B at 64K. At twice its native range, the shipped budget itself binds in the flagged heads. Exact mass at every token of the three flagged heads scores 64.0, level with the repair's 66.0. Full attention in the same heads reads every block and scores 75.5, against 82.5 with full attention in every head (Table 19). The repair thus loses nothing to its 4-bit code there. The budget costs the flagged heads about 11 points, and 7 more lie in the other heads.

Table 18: **One flagged head at many sequences** (H100, one decode step, attention kernels only, microseconds). Full attention's increment over the sparse read of 64 blocks with FlashAttention 3, the repair's grouped 4-bit scan alone, and the scan with the group mass and the top-k. The ratio is full attention's increment over the repair. Above one, the repair is cheaper.

Sequences	Context	Full Attention	Scan	Repair	Ratio
1	32K	8.8	5.3	61.6	0.14
4	32K	26.1	11.2	70.5	0.37
16	32K	82.0	33.7	95.5	0.86
64	32K	302.0	128.7	199.2	1.52
1	64K	16.7	7.5	70.7	0.24
4	64K	50.7	17.7	84.3	0.60
16	64K	169.3	64.7	137.1	1.24
64	64K	645.9	253.9	339.0	1.91
1	128K	32.4	11.3	81.5	0.40
4	128K	95.6	33.5	107.6	0.89
16	128K	341.5	129.1	216.3	1.58
64	128K	1336.2	498.1	604.8	2.21

Table 19: **Where selector repair stops** (MK3, n=200).

System	Arm	Accuracy
NOSA-8B, 32K	Sparse	18.5
	Exact mass at the decode tokens of every head	34.5
	Repair at every token of the five flagged heads	90.0
	Exact mass at every token of the five flagged heads	90.0
	Exact mass at every token of every head	94.0
	Dense in the five flagged heads	94.0
	Dense	94.0
InfLLMv2-8B Retraining, 32K	Sparse	23.5
	Repair at every token of the three flagged heads	84.0
	Exact mass at every token of the three flagged heads	82.5
	Exact mass at every token of every head	85.0
	Dense in the three flagged heads	90.5
	Dense	92.0
InfLLM-V2-8B, 64K	Sparse	16.0
	Sparse, three times the budget, 288 blocks per token	58.0
	Repair at every token of the three flagged heads	66.0
	Exact mass at every token of the three flagged heads	64.0
	Dense in the three flagged heads	75.5
	Dense	82.5

Dash-8B and NSA-8B. Huang et al. (2026) release Dash-8B and a retrained NSA-8B continued from MiniCPM-4-8B. Dash-8B at its 16K envelope scores 99.5 dense, 64.5 shipped and 94.0 with exact mass. Its release gives blocks outside its entmax support a routing prior of $-\infty$, so injected blocks must get a finite prior to be read. NSA-8B scores 13.0 at 16K, against 12.0 in its own report, and 3.0 at 32K. With every block selected through its own kernel, it scores 99.5 and 90.0. At 32K, its shipped selector falls from 95.0 with one needle in noise to 26.5 with three look-alike needles in an essay and 3.0 when every line of the haystack is a look-alike, consistent with the resolution account of Section 5. The repair in every head matches exact mass at every token, 98.0 and 78.5 against 98.5 and 80.0. The three heads flagged on the InfLLM-V2 family give 64.5 and 20.5, so the loss of this release is not confined to those heads. We did not run the probe on it.

H FIVE BENCHMARKS OF REAL DATA

We screened five published benchmarks whose instances have the structure of the tasks the selector fails on, with each benchmark's own data, prompt and metric. We compared full attention with the shipped selector before running any repair, and ran every arm of a cell on one GPU type.

LongFuncEval (Kate et al., 2025), questions over real booking.com API responses on InfLLM-V2-8B. On 450 questions of 5K to 32K tokens, full attention scores 62.9 and the shipped selector scores 58.4. The deficit is carried by the questions that retrieve the record with a given identifier among hundreds of near-identical records, 81.7 against 72.0. It grows with the prompt, 0.0 below 16K, -3.1 from 16K to 24K and -6.7 from 24K to 32K, consistent with Corollary 1. There, the repair recovers 4.4 of the 9.6 points on the identifier questions, and exact mass in every head recovers 7.4. The deficit is thus a selection loss of which the heads calibrated on MK3 reach about half. On the 245 identifier questions of the benchmark's longer split, a median of 54,014 model tokens, full attention scores 64.9 and the shipped selector scores 49.0. The repair at every token of the same three heads scores 59.2, closing two thirds of the gap. A keep-only arm reproduces the shipped selector on every one. On all 464 questions of the longer split, a median of 56K tokens, full attention scores 48.3 and the shipped selector scores 39.7. The question-token variant scores 44.2, half of the gap, and the repair at every token scores 46.1, three quarters of it, with a paired interval against full attention that includes zero. Full attention in the three flagged heads scores 61.3 on the 450 questions, above both repairs. On all 464 of the longer split, it scores 44.2, level with the question-token variant and below the repair at every token, whose paired interval against it excludes zero, since beyond the native range part of the loss lies outside the flagged heads (Appendix G). Recalibrating on 25 of the benchmark's own prompts flags the heads of layers 9, 22, 26 and 29 and scores 54.2. In this one comparison, the heads calibrated on MK3 thus transfer better than heads calibrated on the benchmark itself.

Four benchmarks without a deficit to close. On MiniCPM4.1-8B, LongCodeU (Li et al., 2025) scores 75.1 with full attention against 74.1 shipped, RepoQA (Liu et al., 2024a) scores 84.0 against 82.0 at 16K and 79.0 against 78.4 at 32K, OpenAI MRCR (OpenAI, 2025) scores 19.6 against 19.9, and OOLONG (Bertsch et al., 2025) scores 26.7 against 24.9. The repair stays within two points of the shipped selector on each.

I SCOPE OF THE COMPARISON CLASS AND THE CLOSEST WORK

KV eviction and compression (Zhang et al., 2023; Li et al., 2024; Sun et al., 2025a), with budgets or spans set per head (Fu et al., 2025b; Feng et al., 2025; Fu et al., 2025a), training-free selection for dense backbones (Jiang et al., 2024; Yang et al., 2025b; Hao et al., 2025), retrieval over an offloaded or compressed cache (Liu et al., 2025; Zhang et al., 2025; Chen et al., 2025), serving systems that combine such sparsity (Yang et al., 2025c), and linear, state-space and hybrid architectures (Gu & Dao, 2023; Lieber et al., 2024) change a dense model's cache, compute or architecture. In contrast, we study checkpoints that ship a co-adapted selector at a fixed budget. NSA and MoBA release no trained checkpoints. Token indexers of the DSA kind ship inside models of well over a hundred billion parameters, among them DeepSeek-V3.2 (685B) and GLM-5 (DeepSeek-AI, 2025; GLM-5 Team, 2026), beyond the single-GPU harness we instrument. LongCat-Flash-Lite runs only in a serving stack that our instrumentation does not port to.

Closest work. The deficit itself is on record (Zhao et al., 2026; MiniCPM Team, 2025), but neither report locates it. LOCKS, COBS and BMFA appeared in July 2026, and RBS-Attention, FFD, Fathom and self-indexing attention appeared in September 2026, concurrently with this work. Xiu et al. (2025) measure a trained indexer against exact top-k per head without localizing the loss, and Leng et al. (2026) find that most hierarchical sparse models trained from scratch already retrieve the needle chunk. HISA (Xu et al., 2026) and self-indexing attention (Yang et al., 2026) replace a deployed token indexer without training, in every head and to cut cost. RBS-Attention (Song et al., 2026) names the same failure mean dilution and rescues blocks by a bound on their key radius, in every head of a dense model, to speed up the prefill. Low-bit keys score selections in dense models in SALE (Ji et al., 2026), FFD (Liu et al., 2026b) and Fathom (Kalyanarangan, 2026). The head soft vote is TokenSelect's (Wu et al., 2024a). LOCKS (Hwang, 2026) ranks the pages of dense models by a log-sum-exp mass estimated from compact page-local key codes, the criterion our scorer computes

on a per-channel 4-bit copy of the keys. It shows that averaging the heads' shares maximizes the group's mean captured mass, and proves a shared sketch blind to some differences within a page. COBS (Tian et al., 2026) explains mean-key selectors as the first-order term of a cumulant expansion of a block's log mass. BMFA (Xu & Wong, 2026) proves in vision transformers that a few tokens can keep a block's attention mass while their share of its mean vanishes, the pooled side of Proposition 1. We add to it the exact side's threshold, the competitor spread δ and, with Corollary 1, ranking under a fixed budget. Elastic Attention (Tang et al., 2026), whose InfLLM-V2 checkpoints on Qwen3 we probe, trains a router on a frozen dense backbone that gives about a third of the heads full attention and the rest a sparse pattern, input by input, without attributing any loss to a head. We differ from all of them in locating the failure in a few heads of deployed sparse checkpoints, showing that a retrieval score ranks heads whose selector already finds the needle above them, and repairing it there at the shipped budget without training.