PERISCOPE: EXTENDING FROZEN LANGUAGE MODELS BEYOND THEIR CONTEXT WINDOW

Anonymous authors
Paper under double-blind review

ABSTRACT

A language model reads long text in one forward pass. The pass is quadratic in length, stops at the context window, and loses accuracy with length before reaching it. We ask whether the read can be factorized when it is a decision over a finite set: which document is relevant, which option is supported, which passage is the evidence. **Periscope**, a training-free inference method, arranges the N chunks of a text on a $K \times K$ grid with $K = \lceil \sqrt{N} \rceil$ and asks a frozen model the same question about K local spans of consecutive chunks and K strided spans that sample the whole text, reading the log-odds of every answer at one token. Each answer takes its best local and its best strided score, and scoring every chunk by its two spans gives an evidence map at no further cost, whose peak is the chunk behind the answer. Every probe is about $\sqrt{sc}$ tokens for a text of s tokens and chunk size c, so a window of W tokens reaches W^2/c tokens at $s^{1.5}$ cost. The map replaces the long read. On LongBench v2, reading only the K chunks the map ranks highest, 9k tokens, matches the same model's best window read across windows from 32k to 1M tokens, and on InfiniteBench, where the median context is 150k tokens, it leads the best window read by 5 points. The same map ranks BRIGHT's long-document corpora with the best NDCG@10 of six methods. Each call caches only one probe, so a 27B model reads 4.5M-token contexts on one 80GB GPU, where a single pass would need 296GB of cache. A long read then needs a GPU that holds the model, not one that holds the text.

1 INTRODUCTION

A language model reads long text in a single forward pass, and the pass has three limits. Its attention cost grows with the square of the input. It stops at the context window. And it loses accuracy before it gets there: most long-context models fall below a quality threshold well before their claimed length (Hsieh et al., 2024), 11 of 13 models claiming 128k tokens fall below half of their short-context accuracy by 32k once the needle shares little vocabulary with the question (Modarressi et al., 2025), and accuracy drops with input length even when every relevant passage is present (Du et al., 2025).

Three responses exist, each with a structural limit. Extending the window (Peng et al., 2024; Ding et al., 2024; Jin et al., 2024) moves the wall and keeps the quadratic cost and the monolithic memory of the pass. Reading less, by retrieving chunks with an embedding model (Karpukhin et al., 2020) or compressing the prompt (Jiang et al., 2023), lets another model decide what enters the pass, with no guarantee that what it kept is what the question needs. Reading in pieces bounds every call, but passage scoring sees each piece alone (Dai & Callan, 2019) and a chain of agents passes a lossy summary from one piece to the next (Zhang et al., 2024b), so nothing relates the final answer to a read of the whole text.

We start from what the read is for. Many long-context reads are decisions over a finite set: which document is relevant, which option is supported, which passage is the evidence. Such a decision needs only one supporting passage. A document is relevant if some part of it supports the query, and an option is correct if some passage establishes it. A frozen model can judge, from a bounded window, whether that window supports an answer, and the bounded window is also where the model is accurate. So the text can be covered by bounded windows and the answers composed by a maximum, which adds no approximation of its own, and the answer distribution over the windows is an estimate of where the evidence sits.

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.

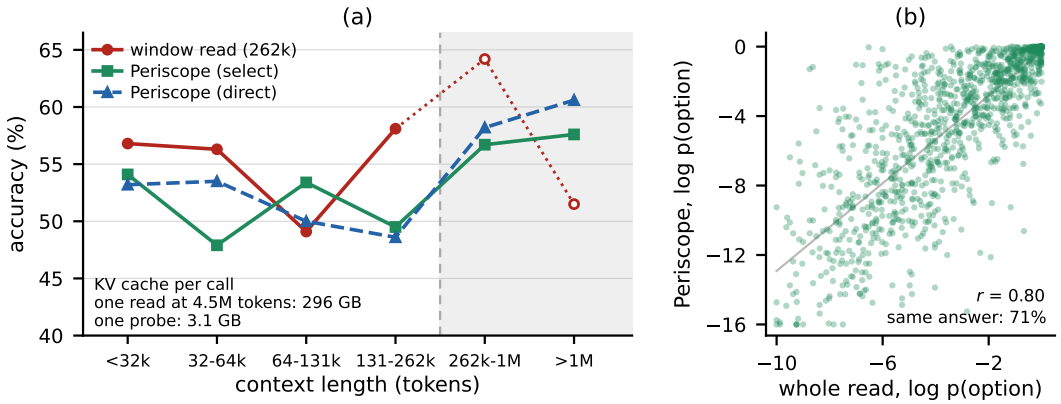

Figure 1: Qwen3.5-27B on LongBench v2. (a) Accuracy by context length. The window read at the native 262k window, dashed, is dotted past it, where it reads only a prefix. (b) The direct readout's log-probability of each option against the whole read's on the 298 contexts read whole.

Periscope (Figure 2) is that read: an inference method with which a frozen model served at a window of W tokens reads texts of up to W^2/c tokens, for chunk size c, with memory bounded per call. It uses the same forward pass and the same one-token readout as the single read, applied to bounded inputs, with nothing generated, no state between calls, and no orchestration. The N chunks of a text are arranged on a $K \times K$ grid with $K = \lceil \sqrt{N} \rceil$. The model answers the same question about K *local* spans, each of K consecutive chunks, and K *strided* spans, each joining every K-th chunk so that it samples the whole text. Every chunk is read twice by $2K$ independent calls of about $\sqrt{sc}$ tokens for a text of s tokens, at a cost growing as $s^{1.5}$. Each answer takes its best local and its best strided score, and scoring every chunk by the two spans it lies in gives a map of where the evidence sits, whose peak is the chunk behind the answer. The same map serves every task through a fixed readout: it ranks documents, answers the question directly, locates the evidence, and selects a read of the K chunks it ranks highest, one probe long.

We evaluate the method as inference, holding the model, the prompt, and the readout fixed and changing only which text reaches each forward pass. On LongBench v2 (Bai et al., 2025), reading the K chunks the map ranks highest, 9k tokens, matches the same model's best window read across windows from 32k to 1M tokens and leads embedding, random, and prefix selection at the same budget by 6 to 11 points. On InfiniteBench En.MC (Zhang et al., 2024a), with a median context of 150k tokens, the same read leads the best window read by 5 points. On BRIGHT (SU et al., 2025) the ranking readout attains the best NDCG@10 and MRR of six methods.

Contributions. (1) *Factorized reading.* A decomposition that captures the context of a whole text through short, independent local and strided probes, so a model with a window of W tokens reads texts of W^2/c tokens without ever reading them at once. (2) *The evidence map.* The probes' answers combine into a map of where the evidence sits in the text, at no extra cost. (3) *Readouts of the map.* The same map ranks documents, answers questions directly, locates the evidence, and selects a short read that replaces reading the whole window.

2 Related Work

Limits of the single long read. Most long-context models fall below a quality threshold well before their claimed length, and only half hold up at 32k tokens (Hsieh et al., 2024), the gap widens once the needle shares little vocabulary with the question (Modarressi et al., 2025), content in the middle of a long input is used less than content at either end (Liu et al., 2024), and accuracy falls with input length even when retrieval is perfect (Du et al., 2025). The limit these studies share is the pass itself: every token competes with every other for the same decision, and the model is most reliable on the short inputs it reads best.

Extending the window. The first response is to make the pass longer. Positional interpolation and its successors extend a pretrained window by orders of magnitude with little or no tuning (Chen et al., 2023; Peng et al., 2024; Ding et al., 2024; Jin et al., 2024), and keep the quadratic attention and the memory of the original. Sub-quadratic architectures address the cost itself. State-space and linear-attention layers carry a fixed-size state (Gu & Dao, 2023; Yang et al., 2025b), and sparse attention reads a selected subset of keys (Liu et al., 2025). A fixed state alone copies and retrieves from context poorly (Jelassi et al., 2024), so deployed models keep a minority of full-attention layers (Lenz et al., 2025; Li et al., 2025), whose key-value cache still grows with every token. The largest of these models need a multi-GPU node for a long read: Jamba-1.5-Large fits a 256k context on eight 80GB GPUs with 8-bit expert weights (Team et al., 2024), and MiniMax-Text-01 was sized to process over 1M tokens on a single eight-GPU, 640GB machine with 8-bit quantization (Li et al., 2025). These are also architectures rather than inference methods, trained or continually pretrained with their layers in place, so they do not apply to an existing model. In every case the whole text enters one pass, and the evaluations above find accuracy declining well inside the window.

Reading less. The second response keeps the pass short by deciding in advance what enters it. Retrieval-augmented generation reads the top-k chunks a retriever selects (Lewis et al., 2020; Karpukhin et al., 2020), and prompt compression drops tokens before the pass (Jiang et al., 2023). Dense and late-interaction retrievers (Karpukhin et al., 2020; Khattab & Zaharia, 2020) score passages from representations computed before the query is seen, and such retrievers perform poorly when relevance requires reasoning rather than surface matching (SU et al., 2025). Memory-augmented models retrieve blocks of the reader's own key-value cache by similarity, and for EM-LLM also by temporal contiguity (Xiao et al., 2024; Fountas et al., 2025). In every case a selection rule fixed before or outside the reader's judgment of the answer decides what is read, and what the selector misses cannot be recovered by the read.

Language models as judges. The reader can make that decision itself. Pointwise rerankers read a relevance score from a language model's one-word answer, trained (Nogueira et al., 2020) or zero-shot, possibly after generated query and document analyses (Niu et al., 2024), and listwise and setwise rerankers read an ordering (Sun et al., 2023; Zhuang et al., 2024), with the cost per judgment accounted in FLOPs by Peng et al. (2025). A judgment read from a single answer token is cheap and reliable, but rerankers assume the document fits in one call and that a first stage assembled the candidates.

Reading in pieces. A text that does not fit one call can be judged in parts. FirstP scores the lead passage and MaxP the best passage of a document (Dai & Callan, 2019), so every call is bounded, but no call sees beyond its own passage. Chain of Agents processes chunks in sequence, each worker passing a generated message to the next and a manager writing the answer (Zhang et al., 2024b). LongAgent has members read chunks in parallel and report to a leader over several rounds, each round depending on the leader's previous state (Zhao et al., 2024). These systems carry information across the text, but their calls depend on earlier generated outputs, each message is an intermediate that can drop what a later chunk would have needed, and the final answer has no stated relation to a read of the full text. Bounded, independent calls that still see across the whole text would need a different way to cut it.

Separable structure and answer-space probing. A matrix built from one vector along its rows and one along its columns is the cheapest structure a matrix can have, and it is used wherever the full matrix is too costly to measure or learn, from separable filters to low-rank weight updates (Hu et al., 2022). Grid probing carried this structure to inference: it places video frames on a $K \times K$ grid, asks a frozen model about each row and each column, and combines the answers into a map that selects frames for a second pass, $2K$ calls for K^2 frames (Eltahir et al., 2026). The number of calls grows with the square root of the number of items rather than with the number itself. A long text has no such geometry of its own. If it could be folded into a separable grid, the same structure might avoid the two bottlenecks of the single read, the quadratic cost of full attention and the fixed boundary of the window.

Together these threads describe what a read past the window needs: calls that are bounded and independent, a decision made by the reader itself, a view of the whole text and not only of its parts, and a score that shows where its evidence lies.

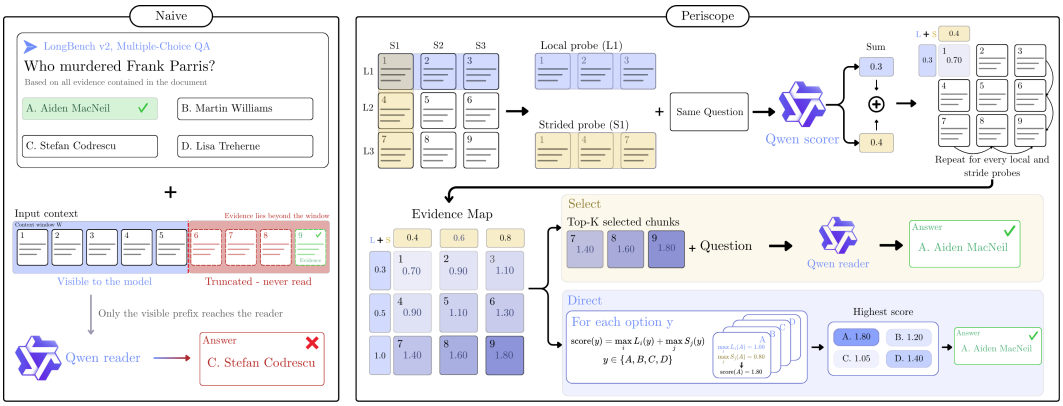

Figure 2: **Periscope** on a LongBench v2 question. Left: the window read sees only the first chunks and misses the evidence in chunk 9. Right: the chunks sit on a $K\times K$ grid, here 3×3, each local probe L_i reads K consecutive chunks and each strided probe S_j every K-th chunk, and each chunk scores the sum of its two probes. The map peaks at the evidence chunk, and the answer comes from reading the top K chunks (Select) or from each option's best local and strided scores alone (Direct).

3 PERISCOPE

Periscope replaces the single forward pass over a text with $2K$ bounded passes and a composition rule. The task determines only the answer set of the probe and which readout of the composed scores is used. The model, its serving, and its readout are those of the single read. Nothing is generated, no state passes between calls, and the calls are independent.

3.1 THE PROBE

A probe is one forward pass of a frozen model θ on a span S of the text and the question q, read at the first answer token. It returns one score per answer, measuring how strongly the span alone supports that answer. For a multiple-choice question with options $\mathcal{Y}$, the prompt lists the options and one more, an abstain option $\bar{y}$. The probe score $\ell(y \mid S, q)$ of each option y is its log-odds against abstaining,

$$\ell(y \mid S, q) \;=\; \log p_\theta(y \mid S, q) \;-\; \log p_\theta(\bar{y} \mid S, q), \tag{1}$$

where $p_\theta(\cdot \mid S, q)$ is the model's probability of each answer at that token. The abstain option lets the model report that the span holds no evidence, so such a span scores every option low rather than favoring one at random, and scores from different spans can be compared. For relevance, the question asks whether S is relevant to a query, and the score is the log-odds of Yes against No.

3.2 LOCAL AND STRIDED SPANS

We split a text x of s tokens into $N{=}\lceil s/c \rceil$ chunks $x_1, \ldots, x_N$ of c tokens, set $K{=}\lceil \sqrt{N} \rceil$, pad to K^2 chunks with empty ones, and index the chunks as a $K\times K$ grid so that chunk $x_{(i-1)K+j}$ occupies cell (i, j). The i-th local span and the j-th strided span are

$$\mathcal{L}_i = \big(x_{(i-1)K+1}, \ldots, x_{iK}\big), \qquad \mathcal{S}_j = \big(x_j,\, x_{j+K}, \ldots, x_{j+(K-1)K}\big), \tag{2}$$

each a concatenation of K chunks in reading order. A *local* probe reads $\mathcal{L}_i$, a stretch of K consecutive chunks. A *strided* probe reads $\mathcal{S}_j$, every K-th chunk, a uniform sample of the whole text. Every chunk lies in exactly one local and one strided span, so it is read twice. Each span holds K chunks of c tokens, about $\sqrt{sc}$ tokens, so every probe is a short call, and the $2K$ probes do not depend on each other. We write $L_i(y)$ for the score of answer y from the i-th local probe and $S_j(y)$ for its score from the j-th strided probe.

3.3 COMPOSITION, MAP, AND READOUTS

An answer is supported by a text if some part of the text supports it, so one strong span is enough. We therefore take, for each answer, its best score in each family and add the two,

$$\text{score}(y) \;=\; \max_i L_i(y) \;+\; \max_j S_j(y). \tag{3}$$

An answer scores high only when some local span and some strided span both support it.

The map. Each chunk lies in one local span $\mathcal{L}_i$ and one strided span $\mathcal{S}_j$, and the two scores of those spans give the chunk a score,

$$M[i,j] \;=\; \max_{y \neq \bar{y}} \big(L_i(y) + S_j(y)\big), \tag{4}$$

where cell (i, j) is chunk $x_{(i-1)K+j}$. The map needs no further calls, and it agrees with the answer: the largest cell for y equals $\text{score}(y)$, so the peak of the map is the chunk whose two spans produced the answer.

Readouts. The same $2K$ probes serve four tasks through four fixed readouts, none of which changes the probes.

- **Ranking.** With $\mathcal{Y} = \{\text{Yes}, \text{No}\}$, $\text{score}(\text{Yes})$ is one number per document, and a corpus is ranked by it.
- **Direct answer.** With options as the answer set, $\arg\max_{y \neq \bar{y}} \text{score}(y)$ answers the question with no further call.
- **Localization.** The peak of M names the chunk that produced the decision.
- **Selected read.** A second stage after the probes. The K chunks with the highest cells of M are joined in their original order, and the model reads them with the question in one more call and answers, as a single read would. The read is Kc tokens, the length of one probe, so the two stages together cost $2K + 1$ probes.

3.4 REACH, COST, AND MEMORY

Reach. We first ask how long a text a model with a window of W tokens can read this way. A probe reads K chunks of c tokens, so it fits the window when $Kc \leq W$, ignoring the short prompt. The largest grid therefore has $K \approx W/c$, and it holds K^2 chunks, $K^2 c$ tokens in all. The longest text the window can read is

$$s_{\max} \;=\; K^2 c \;\approx\; \Big(\frac{W}{c}\Big)^2 c \;=\; \frac{W^2}{c} \tag{5}$$

tokens, W/c times the window itself. At c=500, a 32k window gives $K \approx 65$ and reaches 2M tokens, and a 128k window gives $K \approx 256$ and reaches 33M. This bounds the length a window can read, not the accuracy of the read, which Section 4.3 measures.

Cost. We next compare the compute of the probes with that of a single read of the same text. Let a be the FLOPs a model spends on each token in the layers that process tokens one at a time, which grows with its number of parameters, and b the FLOPs its attention spends on each pair of tokens, which grows with its number of layers and their width. A forward pass over t tokens then costs about $at + bt^2$ FLOPs, since attention compares every token with every other. Ignoring the input (short) prompt, a single read of the text and the $2K$ probes of $\sqrt{sc}$ tokens each cost

$$F_{\text{full}}(s) \;=\; as + bs^2, \qquad F_{\text{probe}}(s) \;=\; 2K\big(a\sqrt{sc} + b\,sc\big) \;=\; 2as + 2b\sqrt{c}\,s^{3/2} \tag{6}$$

using $K = \sqrt{s/c}$. The probes do twice the per-token work, because every chunk is read twice, but their attention cost grows as $s^{1.5}$ rather than s^2. Short texts are therefore cheaper to read whole and long texts cheaper to probe, and the two costs are equal at

$$s^\star \;=\; \big(\sqrt{c} + \sqrt{c + a/b}\big)^2 \;\approx\; a/b, \tag{7}$$

which is tens to hundreds of thousands of tokens for current models.

Memory. A pass stores a key-value cache for every token it reads, so a single read holds the cache of all s tokens, and for long texts that cache outgrows the model's weights. A probe holds the cache of

its $\sqrt{sc}$ tokens only, $K = \sqrt{s/c}$ times less than the single read, and the factor grows with the text: the longer the text, the larger the saving. Because a probe never exceeds the window, its cache never exceeds that of one window, whatever the length of the text. The memory a long read needs is then set by the model's weights, not by the text, and a model that fits a GPU can read texts far longer than that GPU could cache in one pass. Section 4.6 measures the compute, the latency, and the memory of both reads.

4 EXPERIMENTS

4.1 SETUP

All models are frozen, served with vLLM in bfloat16 on one A100 80GB, and read at a single answer token with the settings of Appendix A.1. Every arm runs on the same GPU, so no arm has more memory or compute than another. Every baseline is training-free. Chunks are 500 tokens for question answering and 100 for retrieval.

4.2 BENCHMARKS & BASELINES

Question answering uses LongBench v2 (Bai et al., 2025), 503 four-option questions over contexts of 8k to 2M words in three splits, short (180), medium (215), and long (108), and InfiniteBench En.MC (Zhang et al., 2024a), 229 four-option questions over whole novels with a median context of 150k tokens. The models are Qwen3.5-4B and Qwen3.5-27B (Qwen Team, 2026) and the mixture-of-experts Gemma-4-26B-A4B-it (Gemma Team, 2026). The **window read** reads the longest prefix of the context that fits the window, and we run it at every window the GPU can serve for each model: 32k, 64k, 131k, and 262k natively and 1M with YaRN for the 4B, 131k and 262k for the 27B, and 262k for Gemma. **Periscope** needs far less: its longest probe is 48k tokens, so it runs at every model's native window, without extension. **Periscope** is run in two ways. **Direct** answers from the $2K$ probes alone, with no further call. **Select** has two stages: the probes build the map, and one more call reads the K chunks the map ranks highest, about 9k tokens, and answers. Three baselines replace the map in the second stage and read the same number of chunks: the first K, K at random, and the top K by cosine similarity to the question under Qwen3-Embedding-4B. Every arm answers every question.

Retrieval uses the long-document corpora of BRIGHT (SU et al., 2025), 3,792 documents with medians of 334 to 6,701 tokens by domain and maxima above 1.8 million, and 350 queries stratified sampled across seven of BRIGHT domains, each scored against its whole domain corpus with no candidate pooling. The model is Qwen3-4B-Instruct-2507 (Yang et al., 2025a) at a 16k window, and it scores every document with the relevance probe of Section 3.1. **Periscope** is the ranking readout. The window read has the same model read the longest prefix of each document that fits, as a pointwise LLM reranker does (Niu et al., 2024), FirstP reads the first chunk (Dai & Callan, 2019), and BM25 (Robertson & Zaragoza, 2009) and Qwen3-Embedding-4B, the parameter-matched similarity control, are run under the identical protocol. We report NDCG@10, the benchmark's headline metric, MRR, and recall at 1 and 10.

4.3 THE MAP REPLACES THE READ

A read of K chunks matches the best window read. Across the five windows we run for the 4B, its window read peaks at 131k, at 41.9 on all 503 questions. Reading only the K chunks the map ranks highest, 9k tokens, scores the same 41.9, and every other window scores lower (Table 1). Against this best window, the split falls where the cost model of Section 3.4 puts it: on the short split every context fits and the window read is ahead by 3.9 points, on the medium split, where 44% of contexts exceed 131k tokens, the map is ahead by 3.3, and on the long split, where all of them do, the two tie. By domain (Appendix A.4, Table 6), the selected read leads or ties wherever the evidence sits in one region and trails only on domains where the evidence is spread across documents.

Why a longer window does not help. A longer window lets the model see more of the text, yet the 4B scores 38.2 at 64k, 41.9 at 131k, and then 41.6 at 262k and 39.2 at 1M with YaRN, where 93% of the contexts fit whole. The 1M read drops even on the short split, whose contexts already fit at 131k, so the longer window costs accuracy without adding any text. A probe avoids it because it never sees a long input. It reads about $\sqrt{sc}$ tokens, 9k on average and 48k at the longest context,

Table 1: LongBench v2 accuracy (%) by split and the cost of each arm. *Read*: tokens in the final call, K chunks being 9k on average. *PFLOPs*: mean compute per question from the exact token counts of every call. *KV*: peak key-value cache of one call, GB. A dash: the read does not fit one A100 80GB. Bold: best per column within a block.

Method	Read	PFLOPs	KV	Short	Medium	Long	All
(a) Qwen3.5-4B							
Window read, 32k	32k	0.3	1.1	41.1	34.0	33.3	36.4
Window read, 64k	64k	0.6	2.1	**45.6**	33.5	35.2	38.2
Window read, 131k	131k	1.2	4.3	45.0	38.6	**43.5**	**41.9**
Window read, 262k	262k	2.4	8.6	**45.6**	37.2	**43.5**	41.6
Window read, 1M (YaRN)	1M	8.4	32.8	43.9	34.4	40.7	39.2
Window read, whole context	4.5M	–	148	–	–	–	–
First K chunks	K chunks	0.1	1.6	30.0	29.8	36.1	31.2
Random K chunks	K chunks	0.1	1.6	36.7	30.7	31.5	33.0
Embedding, K chunks	K chunks	1.9	1.6	36.7	33.0	39.8	35.8
Periscope, direct	none	4.5	1.6	43.3	36.7	41.7	40.2
Periscope, select	K chunks	4.6	1.6	41.1	**41.9**	**43.5**	**41.9**
(b) Qwen3.5-27B							
Window read, 131k	131k	6.3	8.6	57.2	52.1	50.0	53.5
Window read, 262k	262k	10.8	17.2	**57.8**	**53.5**	**57.4**	**55.9**
Window read, 1M (YaRN)	1M	–	65.5	–	–	–	–
Window read, whole context	4.5M	–	296	–	–	–	–
Embedding, K chunks	K chunks	2.3	3.1	46.1	40.5	54.6	45.5
Map of the 4B, K chunks	K chunks	5.1	3.1	51.1	45.6	52.8	49.1
Periscope, direct	none	29.7	3.1	54.4	48.8	**57.4**	52.7
Periscope, select	K chunks	30.2	3.1	52.8	**53.5**	50.9	52.7
(c) Gemma-4-26B-A4B-it							
Window read, 262k	262k	2.8	7.9	**43.3**	**43.7**	40.7	42.9
Window read, whole context	4.5M	–	136	–	–	–	–
Periscope, direct	none	4.9	1.5	41.1	38.6	45.4	40.9
Periscope, select	K chunks	5.0	1.5	42.2	**43.7**	**46.3**	**43.7**

so the factorized read stays in the range where the model is accurate. Figure 1(a) shows where this matters for the 27B, against its best window read, at the native 262k. Inside that window, where the whole context is read, the window read leads in three of four bins. Between 262k and 1M tokens it still leads, since its prefix covers at least a quarter of each context. Past 1M tokens the prefix covers less, and on these 33 questions the 262k read scores 51.5, while the map's read scores 57.6 and the direct readout 60.6.

The gain comes from the map. Matching the best window read could come from the budget alone, if any K chunks were enough. It does not: at the same budget, the K chunks the map ranks highest score 6 points above the chunks most similar to the question, 9 above random chunks, and 11 above the first chunks, on every split. The map also picks the chunks the whole read relies on. On the 298 contexts that fit the 131k window, reading the map's chunks gives the same answer as reading the whole context on 71% of questions, against 64% for embedding selection and 58% for random chunks. Even with no read at all, the 27B's direct readout tracks the whole read, its option scores correlating with the whole read's at r=0.80 (Figure 1(b)).

A stronger model gains past the window. The 27B fits one A100 at 131k and 262k, and not at 1M. Its best window read, at 262k, scores 55.9 on all questions, 3.2 points above either readout of **Periscope**, within a point of its 131k read. The two meet on the long split, where every context exceeds 131k tokens. There the direct readout, answering from the probes with no read at all, scores 57.4, the same as the 262k read and 7.4 points above the 131k read, where the 4B's direct readout does not gain. The 262k read reaches the same accuracy only by holding 17 GB of cache per call, against 3.1 for a probe.

The result carries across models. The map need not come from the reader. Chosen by the 4B and read by the 27B, it leads embedding selection by 4 points, so a small model can choose what a large one reads. Nor is the result specific to the hybrid attention of the Qwen models. With

Table 2: InfiniteBench En.MC, Qwen3.5-4B, 229 questions, median context 150k tokens.

Method	Read	PFLOPs	KV	Acc.
Window read, 131k	131k	1.8	4.3	74.2
Window read, 262k	262k	3.3	8.6	77.3
First K chunks	K chunks	0.1	0.6	44.1
Random K chunks	K chunks	0.1	0.6	42.8
Periscope, direct	none	2.9	0.6	80.8
Periscope, select	K chunks	3.0	0.6	**82.1**

Table 3: BRIGHT long-document retrieval, full corpus, 350 queries over 3,792 documents, one frozen model (Qwen3-4B) for FirstP, the window read, and **Periscope**. *Window read*: the longest prefix that fits 16k tokens. *Chunk-max*: every 512-token chunk embedded, a document scored by its best chunk. *R@k*: recall at k.

Method	NDCG@10	MRR	R@1	R@10
BM25	.219	.233	.079	.318
Qwen3-Embed-4B	.461	.450	.213	**.671**
Qwen3-Embed-4B, chunk-max	.422	.401	.208	.615
FirstP	.222	.240	.118	.300
Window read, 16k	.453	.453	.225	.633
Periscope	**.493**	**.491**	**.266**	.666

Gemma-4-26B-A4B-it, a mixture of experts served at its native 262k, the selected read leads the window read, 43.7 against 42.9, ties it on the medium split, and leads by 5.6 points on the long split, with a fifth of the cache.

4.4 A SECOND BENCHMARK

On InfiniteBench the contexts are whole novels, and two thirds of them exceed the 131k window. The map's read of K chunks scores 82.1, 5 points above the best window read, at 262k, and 8 above the read at 131k (Table 2). The gain comes from the novels that do not fit: on the 79 that fit 131k tokens the 131k read is ahead, 82.3 against 78.5, as on the short split of LongBench v2, and on the 150 that do not the map's read leads by 14 points, 84.0 against 70.0.

4.5 IS THE MAP RIGHT?

The probe scores must also be right on their own, apart from the read that follows them. Table 3 tests the ranking readout, with no read at all, on BRIGHT, where relevance takes reasoning rather than word overlap. The probes rank the corpora with the best NDCG@10 and MRR of six methods. Against the embedder, a retriever trained for the task, the frozen model leads MRR and Recall@1, and trails by half a point on Recall@10: the verdict is sharper at the head of the ranking, while similarity fills the top ten more completely.

4.6 COST

The probes process every token twice in calls of about $\sqrt{sc}$ tokens, so on LongBench v2 they process 2.3 and 2.5 times the tokens of the window read on the short and medium splits and 14 times on the long split. Whether that is more compute than the window read depends on the crossover of Section 3.4, which is 119k tokens for the 4B and 278k for the 27B from their measured coefficients (Figure 3(b), Appendix A.2). Measured on one A100, a batch of the 4B's probes takes 26 s at 262k tokens against 40 s for the single read, and 108 s at 1M against 457 s (Figure 3(a)). Memory does not depend on the crossover. At the longest context of the benchmark, 4.5M tokens, one probe of the 27B holds 3.1 GB of cache next to 54 GB of weights, where a single read would hold 296 GB, more than three A100s, so the 27B reads it on one GPU. The 4B served with only 16 GB caches at most 178k tokens in one pass, and its single read ends there while every probe still runs. Past the window, the single read is not available at any price.

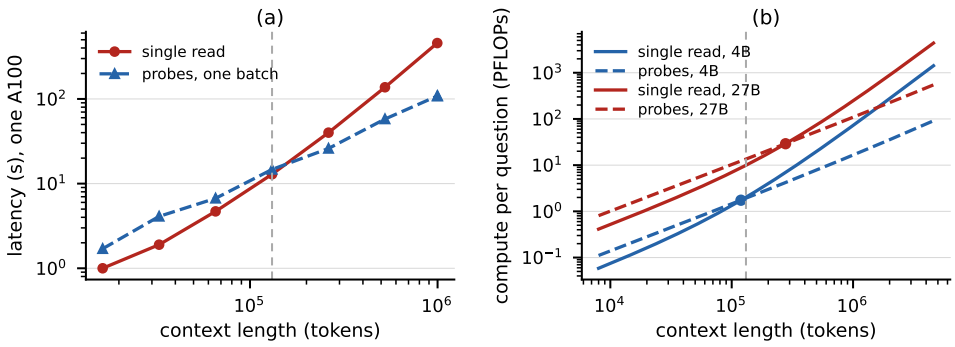

Figure 3: When do the probes cost less than the read? (a) Wall-clock of one single read against one batch of its $2K$ probes, Qwen3.5-4B on one A100 served at 1M tokens. (b) Compute of both arms from Eq. 6 with measured coefficients, crossovers marked.

Table 4: Ablations. (a) Local probes with and without the strided probes, direct readout, with BRIGHT NDCG@10 on 150 queries. (b) Independent probes against a chain over the same K local spans, with latency per question, LongBench v2, Qwen3.5-4B.

(a) Local and global views

Probes	LongBench v2	BRIGHT
one per chunk (MaxP)	–	.551
local only	39.4	.551
local + strided (default)	**40.2**	**.637**

(b) Reading in pieces

Method	Acc.	Latency
Chain of Agents	32.0	207 s
Periscope, selected read	**41.9**	**63 s**

5 Ablations

We run two ablations with the probes held fixed: what the strided probes add, and how the probes compare with reading in sequence.

The strided probes supply the global view. Each local probe sees one neighborhood of the text, and each strided probe a sample of all of it, so removing the strided probes isolates what the global view adds (Table 4a). On BRIGHT, reading each chunk alone, as MaxP does, and reading the local spans give the same NDCG@10, .551, so finer local pieces add nothing. Adding the strided probes lifts it to .637. What the local view lacks is not resolution but a view of the whole document, and strided probes supply it with that. On LongBench v2 the strided probes add 0.8 on all questions.

Independent probes or a chain. Chain of Agents (Zhang et al., 2024b), run with the same model, and the same K local spans, scores 10 points below the selected read, at three times the latency (Table 4b). Its $K{+}1$ calls run in sequence, and each message is written before the next span is read, so what a later span would have made relevant may already be gone.

6 Conclusion and Limitations

We asked whether a long read can be factorized when it is a decision. **Periscope** reads a text through $2K$ bounded local and strided probes of a frozen model, reaching W^2/c tokens from a window of W with the cache of one probe, while achieving a comparable performance of full read.

Limitations and future work. The method covers decisions over a finite answer set. It joins evidence from distant spans only in the selected read, which trails the window read on multi-document QA, and below the crossover it costs more than a single read. Two directions follow: readouts for open-ended outputs such as summaries, and distilling the probe scores and maps into retrievers and rankers far cheaper than the model that produced them.

REPRODUCIBILITY STATEMENT

All experiments use open-weight models served with vLLM in bfloat16 on the public benchmarks LongBench v2, InfiniteBench, and BRIGHT, with every call on a single A100 80GB (Section 4.1). Every prompt is listed verbatim in Appendix A.1 with the readout and sampling settings, the cost coefficients and their derivation are in Appendix A.2, and the run-to-run variation of batched inference is measured in Appendix A.4. The code, the prompts, and the per-question outputs of every run will be released upon publication.

AI USE STATEMENT

We used generative AI tools to find and screen related work, to discuss experimental design and analyses during the research, to write and debug experiment code, and to draft and polish sections of the text. All AI-assisted code was reviewed by the authors and verified against independent runs, all numbers were produced by our own executions on our own hardware, all citations were manually verified against their sources, and every claim and interpretation was decided by the authors. We take responsibility for the final content of this work.

REFERENCES

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3639–3664, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.183. URL https://aclanthology.org/2025.acl-long.183/.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023.

Zhuyun Dai and Jamie Callan. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pp. 985–988, 2019.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longrope: extending llm context window beyond 2 million tokens. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Babu Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. Context length alone hurts LLM performance despite perfect retrieval. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23281–23298, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.1264. URL https://aclanthology.org/2025.findings-emnlp.1264/.

Mohamed Eltahir, Lama Ayash, Ali Habibullah, Tanveer Hussain, and Naeemullah Khan. Gridprobe: Posterior-probing for adaptive test-time compute in long-video vlms, 2026. URL https://arxiv.org/abs/2605.10762.

Zafeirios Fountas, Martin A Benfeghoul, Adnan Oomerjee, Fenia Christopoulou, Gerasimos Lampouras, Haitham Bou Ammar, and Jun Wang. Human-inspired episodic memory for infinite context llms. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 77230–77268, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/c05144b635df16ac9bbf8246bbbd55ca-Paper-Conference.pdf.

Gemma Team. Gemma 4 model card. https://ai.google.dev/gemma/docs/core/model_card_4, 2026. Google DeepMind.

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

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.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 21502–21521. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/jelassi24a.html.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13358–13376, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.825. URL https://aclanthology.org/2023.emnlp-main.825/.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. LLM maybe LongLM: SelfExtend LLM context window without tuning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22099–22114. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/jin24b.html.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 39–48, 2020.

Barak Lenz, Opher Lieber, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, Daniel Gissin, Daniel Jannai, Dor Muhlgay, Dor Zimberg, Edden Gerber, Elad Dolev, Eran Krakovsky, Erez Sa, Erez Schwartz, Gal Cohen, Gal Shachaf, Haim Rozenblum, Hofit Bata, Ido Blass, Inbal Magar, Itay Dalmedigos, Jhonathan Osin, Julie Fadlon, Maria Rozman, Matan Danos, Michael Gokhman, Mor Zusman, Naama Gidron, Nir Ratner, Noam Gat, Noam Rozen, Oded Fried, Ohad Leshno, Omer Antverg, Omri Abend, Or Dagan, Orit Cohavi, Raz Alon, Ro'i Belson, Roi Cohen, Rom Gilad, Roman Glozman, Shahar Lev, Shai Shalev-Shwartz, Shaked Meirom, Tal Delbari, Tal Ness, Tomer Asida, Tom Ben Gal, Tom Braude, Uriya Pumerantz, Joshua Cohen, Yonatan Belinkov, Yuval Globerson, Yuval Levy, and Yoav Shoham. Jamba: Hybrid transformer-mamba language models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 67959–67984, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/a9ed43fa31dc8b4a7d7a673d713dcb5f-Paper-Conference.pdf.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In

H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schuetze. NoLiMa: Long-context evaluation beyond literal matching. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 44554–44570. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/modarressi25a.html.

Tong Niu, Shafiq Joty, Ye Liu, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. Judgerank: Leveraging large language models for reasoning-intensive reranking. *arXiv preprint arXiv:2411.00142*, 2024.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 708–718, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.63. URL https://aclanthology.org/2020.findings-emnlp.63/.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 31932–31951, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/874a4d89f2d04b4bcf9a2c19545cf040-Paper-Conference.pdf.

Zhiyuan Peng, Ting-Ruen Wei, Tingyu Song, and Yilun Zhao. Efficiency-effectiveness reranking FLOPs for LLM-based rerankers. In Saloni Potdar, Lina Rojas-Barahona, and Sebastien Montella (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 2782–2791, Suzhou (China), November 2025. Association for Computational Linguistics. ISBN 979-8-89176-333-3. doi: 10.18653/v1/2025.emnlp-industry.186. URL https://aclanthology.org/2025.emnlp-industry.186/.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 4(1-2):1–174, 2009.

Hongjin SU, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Liu Haisu, Quan Shi, Zachary Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan Arik, Danqi Chen, and Tao Yu. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 48941–48991, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/7a0f8055c838df8e62329a76c7c6403d-Paper-Conference.pdf.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? investigating large language models as re-ranking agents. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14918–14937, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.923. URL https://aclanthology.org/2023.emnlp-main.923/.

Jamba Team, Barak Lenz, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, et al. Jamba-1.5: Hybrid transformer-mamba models at scale. *arXiv preprint arXiv:2408.12570*, 2024.

Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. Infllm: Training-free long-context extrapolation for llms with an efficient context memory. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 119638–119661. Curran Associates, Inc., 2024. doi: 10.52202/079017-3801. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/d842425e4bf79ba039352da0f658a906-Paper-Conference.pdf.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 29687–29707, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/4904fad153f6434a7bcf04465d4be2cc-Paper-Conference.pdf.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, et al. ∞bench: Extending long context evaluation beyond 100k tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15262–15277, 2024a.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Ö. Arı k. Chain of agents: Large language models collaborating on long-context tasks. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 132208–132237. Curran Associates, Inc., 2024b. doi: 10.52202/079017-4202. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ee71a4b14ec26710b39ee6be113d7750-Paper-Conference.pdf.

Jun Zhao, Can Zu, Hao Xu, Yi Lu, Wei He, Yiwen Ding, Tao Gui, Qi Zhang, and Xuanjing Huang. Longagent: scaling language models to 128k context through multi-agent collaboration. *arXiv preprint arXiv:2402.11550*, 2024.

Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. A setwise approach for effective and highly efficient zero-shot ranking with large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 38–47, 2024.

A Appendix

A.1 Implementation

Prompts. Every call is one user turn of the model's chat template. Both the relevance and question probes place the text first so that it is a shared prefix across queries or questions. The relevance probe is

```
Document: {span}
Query: {query}
Is this document relevant to the query? Answer Yes or No:
```

with No as the abstain option. The question probe is

```
{span}

Question: {question}
A. {option A}
B. {option B}
C. {option C}
D. {option D}
E. Unsure
Answer with a single letter:
```

with `E. Unsure` as the abstain option $\bar{y}$. The selected read and the window read use the same prompt without the line `E. Unsure`. Each call is one forward pass with `max_tokens=1`. We read the top-20 log-probabilities at the answer position and compute Eq. 1. Thinking mode is disabled where the model has one.

Sampling. Temperature 1.0 with `top_p`$= 1$, `top_k`$= -1$, and `min_p`$= 0$. Temperature rescales every log-odds by the same constant and cannot change a ranking. Nucleus and top-k truncation can remove an answer token from the returned distribution, so they are disabled.

Answer tokens. The surface form of the answer token (`Yes` against `␣Yes`) depends on model and template. We take the maximum over surface variants and verify at startup on one probe with a known answer.

Chunking and grid. Texts are tokenized once and split into c-token chunks. $K{=}\lceil\sqrt{N}\rceil$ and the grid is padded with empty chunks that are dropped from the spans, so no chunk is discarded, and a span that would be empty is not issued. The chunks of a span are joined by a single space with no delimiter token, so a strided span reads as running text with a discontinuity at each chunk boundary. Scores are used as read, with no calibration across spans. Chunk lists are cached per document.

Serving. Each model is served with vLLM on a single A100 80GB GPU, the 4B at 32 concurrent sequences and the 27B at 8, and every arm of that model, the window read included, runs against the same instance. The $2K$ probes of a text are issued as one batch. The window read takes the longest prefix whose prompt fits the window and records the truncation, for retrieval and for question answering alike. Every arm uses the same first-token letter readout, and no question is defaulted to a letter.

Baselines. BM25 uses $k_1{=}0.9$ and $b{=}0.4$. Qwen3-Embedding-4B uses last-token pooling with L_2 normalization and the standard retrieval instruction on the query side. For retrieval, documents are truncated to 2,048 tokens and scored by cosine similarity over the full corpus. For chunk selection on LongBench v2, every 500-token chunk is embedded and the top K by cosine similarity to the question are read in original order. Random selection uses a fixed seed. Chain of Agents (Zhang et al., 2024b) uses the worker and manager prompts of that paper's Table 9 verbatim: the workers read the K local spans of the grid in order, the same spans as the local probes, each writing a summary of at most 512 tokens that carries the evidence to the next, and the manager answers from the last summary with the letter readout of every other arm, $K{+}1$ calls per question. Summaries are decoded greedily.

A.2 COST

Full cost with the prompt. With a prompt of p tokens, the single read costs $F_{\text{full}}(s) = a(s+p) + b(s+p)^2$ and the $2K$ probes of $\sqrt{sc}+p$ tokens cost

$$F_{\text{probe}}(s) \;=\; 2as + 2b\sqrt{c}\,s^{3/2} + 4bps + \left(2ap + 2bp^2\right)\sqrt{s/c}.$$

The prompt terms are lower order in s, and the crossover of Eq. 7 drops them.

Per-call compute is $2P_{\text{ne}}t + 2Ldt^2$ FLOPs for a prompt of t tokens, with P_{ne} the non-embedding parameter count, L the number of layers, and d the hidden size. For Qwen3-4B-Instruct-2507, P_{ne}=3.633 $\times 10^9$, L=36, d=2560, and the form matches hardware FLOP counters for single passes on this model. Token counts are measured on the exact prompt, template included. The FLOPs crossover of Section 3.4 is the positive root of $F_{\text{full}} = F_{\text{probe}}$, which with these coefficients and c=100 is 65.9k tokens for this dense model, and 15.7k against the local probes alone. In the notation of Section 3.4, $a = 2P_{\text{ne}} = 7.3\times10^9$ FLOPs per token, the attention coefficient $b = 2Ld = 1.8\times10^5$, and p is the measured median prompt length per domain.

Measured latency. Figure 3(a) in the main text is measured on one A100 80GB with vLLM, Qwen3.5-4B served at 1M tokens with YaRN, one single read against one batch of the $2K$ probes of the same text, random unique text so that prefix caching cannot help, both arms on the same idle server within the same minute. The single read takes 1.0, 1.9, 4.7, 12.9, 40, 137, and 457 s at 16k, 32k, 64k, 131k, 262k, 524k, and 996k tokens, the probes 1.7, 4.1, 6.7, 14.7, 26, 58, and 108 s in 12 to 90 calls. The probes cost more time up to 131k, 1.1 times there, and less from 262k on, 0.65 times at 262k and 0.24 at 1M. The longest single read this server completes is 996,220 tokens, the model length it was served at. Its cache would hold 2.1M.

Qwen3.5-4B and Qwen3.5-27B are hybrid models: 8 of 32 layers and 16 of 64 layers attend with full attention (hidden sizes 2,560 and 5,120, 16 and 24 attention heads of dimension 256, 4 key-value heads), and the remaining layers are linear in sequence length. The attention coefficient counts only the full-attention layers, $b = 2L_{\text{attn}} \times \text{heads} \times 256$, which is 6.6×10^4 for the 4B and 2.0×10^5 for the 27B, against $a = 2P_{\text{ne}}$ of 6.8×10^9 and 5.0×10^{10}. The crossover of Section 3.4 at c=500 is then 119k tokens for the 4B and 278k for the 27B. Below it the probes cost more than the single read, at 131k about the same for the 4B and 1.4 times for the 27B. Above it the ratio falls as $s^{-1/2}$: at 262k, the native window of these models, the probes cost 0.63 and 1.03 times the single read, at 1M tokens 0.23 and 0.44 times, and at the longest LongBench v2 context, 4.5M tokens, 0.07 and 0.13 times. On LongBench v2 the probes' compute tracks the token ratios of Section 4.6: 2.3 and 2.5 times the window read on the short and medium splits, and 14 times on the long split, where the window read stops at 131k tokens and the probes read the whole context. The exact prompt token counts of every call are stored with the results.

Memory. A full-attention layer stores keys and values for every token of a call, $2 \times 4 \times 256 \times 2$ bytes per token per layer with 4 key-value heads of dimension 256 in bfloat16, which is 64 KB per token over the 27B's 16 full-attention layers and 32 KB over the 4B's 8. The linear layers keep a state of fixed size. One read of the longest LongBench v2 context, 4,523,227 tokens, would therefore hold 296 GB of key-value cache for the 27B and 148 GB for the 4B. A probe holds at most $\sqrt{sc}$ tokens, 47.6k at that context, which is 3.1 GB and 1.6 GB. The serving engine's own accounting agrees: on one A100 80GB, vLLM allocates a cache of 2.1M tokens for the 4B and, next to the 27B's 54 GB of weights, 356k tokens for the 27B. A single read of the 27B past 356k tokens therefore does not exist on this hardware at any positional scaling, which covers 86 of the 108 long questions, and a single read of the 4B stops at 2.1M, short of 14 of them. The wall follows the budget: the same 4B served on the same card with vLLM capped to 16 GB holds 178k tokens (0.68 of the 262k it was configured for, in the engine's own report), completes reads of 131k and 160k tokens in 12.9 and 17.6 s, its longest completed read is 177,500 tokens and a read of 180,000 is never scheduled, and it still runs every probe, since none exceeds 47.6k tokens: the 46 probes of a 262k text finish in 24 s on that budget. At that budget the single read reaches 178k tokens and the probes $178\text{k}^2/c = 63$M, the whole of LongBench v2 included. Reach grows with the square of the cache a device holds, which is what puts the method on small GPUs. A window in Section 4.1 is the context the server is configured for, not the point at which memory runs out. The point is that the probes' memory is bounded by $\sqrt{sc}$ whatever the context, and the single read's is not.

On BRIGHT most documents sit below the crossover and **Periscope** costs 2.6 to 27 times the FLOPs of the whole read by domain, a median of 5. The whole read fails on 0.7 to 26.1% of documents by domain at the 16k window, and on 29% overall in a companion run at a smaller window on 16GB-class hardware. The window is a property of budget and hardware and is reported with every such number.

A.3 Per-Domain Retrieval Results

Table 5: Per-domain NDCG@10, MRR, and Recall@10 on BRIGHT long-document retrieval, full corpus, 50 queries per domain. Bold: best per row.

Domain	BM25	Embed	Embed, chunk-max	FirstP	Window read	**Periscope**
NDCG@10						
Biology	.206	.565	.514	.508	.622	**.688**
Earth science	.184	**.577**	.512	.301	.441	.426
Economics	.239	.464	**.471**	.263	.441	.455
Pony	.355	.143	.069	.099	.273	**.379**
Psychology	.184	.561	.551	.160	.532	**.613**
Robotics	.146	**.402**	.317	.073	.401	.355
Sustainable living	.220	.513	.521	.152	.463	**.533**
Mean	.219	.461	.422	.222	.453	**.493**
MRR						
Biology	.196	.535	.473	.527	.587	**.675**
Earth science	.168	**.579**	.492	.273	.426	.396
Economics	.228	.388	.401	.253	.385	**.408**
Pony	.530	.336	.157	.199	.515	**.576**
Psychology	.157	.521	.522	.162	.493	**.597**
Robotics	.137	.316	.273	.088	**.349**	.298
Sustainable living	.212	.477	**.489**	.181	.417	.485
Mean	.233	.450	.401	.240	.453	**.491**
Recall@10						
Biology	.318	.830	.760	.637	.803	**.837**
Earth science	.323	**.773**	.727	.493	.627	.638
Economics	.320	**.740**	.720	.350	.660	.660
Pony	.394	.145	.079	.122	.286	**.416**
Psychology	.341	**.772**	.748	.227	.721	.755
Robotics	.230	**.710**	.540	.090	.630	.580
Sustainable living	.298	.730	.732	.184	.706	**.776**
Mean	.318	**.672**	.615	.300	.633	.666

Corpus sizes are 508 to 601 documents per domain. The map leads the window read on NDCG@10 and MRR in five domains and trails it in Earth science and Robotics, where the documents that fit the 16k window carry most of the relevance. Against the embedder the map leads MRR in five domains, Recall@1 in six, and Recall@10 in three: the verdict places a relevant document first more often, and similarity brings more of the several relevant documents into the top ten. The largest margin is on Pony, whose queries are code and where similarity to the query text is least informative.

A.4 LongBench v2 Details

Single-document QA and dialogue place the evidence in one region, and there the selected read leads or ties at both scales. In-context learning spreads its demonstrations through the text, and there the 4B trails the window read by 5 points and the 27B leads by 5. Structured data and code reward the document-wide view of the strided probes, and the 27B's beats the window read on both. Multi-document QA, whose evidence lies in different documents, is the one domain behind at both scales, by 6 and 9 points.

A read that cannot fit the window is counted as unanswered and excluded from the accuracy rather than scored as wrong, which occurred for 11 questions at $4K$ chunks, whose accuracy is over the 492 answered, and for none of the arms in Table 1.

Table 6: LongBench v2 accuracy (%) by domain, all splits, both models. Bold: best per row.

Domain	n	Window read	Direct readout	Selected read
Qwen3.5-4B				
Single-document QA	175	38.9	41.7	**43.4**
Multi-document QA	125	**40.0**	32.0	33.6
Long in-context learning	81	**54.3**	43.2	49.4
Long-dialogue history	39	43.6	51.3	**53.8**
Code repository	50	34.0	**38.0**	36.0
Long structured data	33	**45.5**	**45.5**	42.4
Qwen3.5-27B				
Single-document QA	175	**50.9**	48.6	**50.9**
Multi-document QA	125	**53.6**	46.4	44.8
Long in-context learning	81	59.3	63.0	**64.2**
Long-dialogue history	39	53.8	**61.5**	53.8
Code repository	50	54.0	**58.0**	56.0
Long structured data	33	51.5	54.5	**57.6**

Table 7: Questions won and lost against the 131k window read of Qwen3.5-4B on the same questions. n=503 unless stated.

Arm	Wins	Losses	n
Periscope, selected read	54	54	503
Periscope, direct readout	51	60	503
First K chunks	39	93	503
Random K chunks	44	89	503
Embedding K chunks	51	82	503
$K/2$ chunks	52	63	503
$2K$ chunks	40	51	503
$4K$ chunks	40	46	492
Local probes only, direct	52	65	503
Map → Qwen3.5-27B	87	51	503
Embedding → Qwen3.5-27B	92	74	503
Qwen3.5-27B window read	106	48	503
Against the Qwen3.5-27B window read			
Map from Qwen3.5-27B → Qwen3.5-27B, medium	18	15	215
Map from Qwen3.5-27B → Qwen3.5-27B, all	43	47	503
27B direct readout, all	48	52	503
27B direct readout, long	13	5	108
Against the 4B reading 1M tokens with YaRN			
Periscope, selected read	58	44	503
Periscope, direct readout	54	49	503

Chunk size. Doubling the chunk to 1,000 tokens cuts the number of probes by a factor of $\sqrt{2}$ and lengthens each by the same factor, and the selected read on the medium split falls from 41.9 to 38.6, so the map is sharper with the smaller chunk. This difference is close to the run-to-run variation above.

Table 8: Accuracy (%) by context length in tokens, both models, with bin edges at the 131k and 262k windows. Figure 1 plots the 27B's native 262k read and both readouts. A window read past its window reads the longest prefix that fits.

Model	Arm	<32k n=111	32–64k n=71	64–131k n=116	131–262k n=105	262k–1M n=67	>1M n=33
Qwen3.5-4B	Window read, 32k	43.2	39.4	29.3	37.1	32.8	36.4
Qwen3.5-4B	Window read, 64k	43.2	46.5	29.3	35.2	38.8	42.4
Qwen3.5-4B	Window read, 131k	44.1	43.7	34.5	43.8	44.8	45.5
Qwen3.5-4B	Window read, 262k	43.2	46.5	34.5	40.0	49.3	39.4
Qwen3.5-4B	Window read, 1M (YaRN)	41.4	45.1	29.3	41.0	40.3	45.5
Qwen3.5-4B	**Periscope**, selected read	45.9	32.4	39.7	42.9	49.3	39.4
Qwen3.5-4B	**Periscope**, direct readout	45.0	39.4	37.1	37.1	41.8	42.4
Qwen3.5-27B	Window read, 131k	55.9	56.3	50.0	53.3	59.7	39.4
Qwen3.5-27B	Window read, 262k	56.8	56.3	49.1	58.1	64.2	51.5
Qwen3.5-27B	**Periscope**, selected read	54.1	47.9	53.4	49.5	56.7	57.6
Qwen3.5-27B	**Periscope**, direct readout	53.2	53.5	50.0	48.6	58.2	60.6

Table 9: Does the arm make the whole read's decision? The 298 LongBench v2 contexts the 131k window reads whole. *Agree*: same answer as the whole read. r: Pearson correlation of the option log-probabilities with the whole read's, log-softmaxed over the four letters, 1,192 option points. ρ: mean per-question Spearman. The ceiling row is the whole read against a second run of itself in another session, on the 120 medium contexts read whole in both.

Model	Arm	Agree (%)	r	ρ	Acc. (%)
Qwen3.5-4B	**Periscope**, selected read	**70.5**	0.82	**0.70**	40.3
Qwen3.5-4B	**Periscope**, direct readout	67.8	**0.82**	0.68	**40.6**
Qwen3.5-4B	Embedding K chunks	64.4	0.76	0.60	36.2
Qwen3.5-4B	Random K chunks	58.4	0.71	0.60	33.6
Qwen3.5-4B	First K chunks	59.7	0.67	0.58	30.9
Qwen3.5-4B	Whole read, re-run of itself (ceiling, n=120)	95.0	1.00	–	–
Qwen3.5-27B	**Periscope**, selected read	**73.2**	**0.82**	**0.70**	**52.3**
Qwen3.5-27B	**Periscope**, direct readout	71.1	0.80	0.67	52.0
Qwen3.5-27B	Map of the 4B, K chunks	67.1	0.80	0.68	47.3
Qwen3.5-27B	Embedding, K chunks	65.1	0.76	0.62	42.3

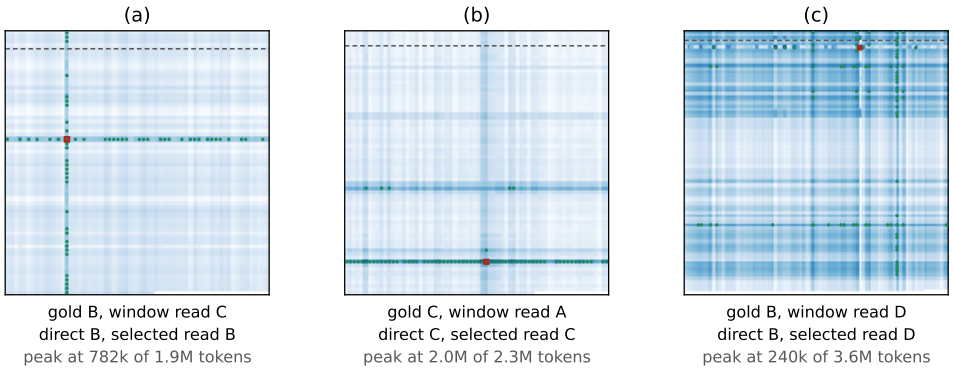

Figure 4: What does the map look like? Three long-split questions, Qwen3.5-27B. Each horizontal line of cells is one local span and each vertical line one strided span, so the document reads line by line. The outlined cell is the peak, the dots are the K selected chunks, and the dashed line marks the last local span a 131k read sees. (a) The peak's local and strided spans cross on the chunk that holds the answer, 782k tokens in, and the selection reads that local span. (b) The peak is 2.0M tokens into a price table, and the selection reads its local span. (c) A flat map, its peak 0.4 above the next cell: the selection scatters and the selected read misses, while the direct readout, which uses only the best score of each answer, still answers right.