

UNDERSTANDING IS DONE EARLY: A DEPTH DIVISION OF LABOR IN LARGE LANGUAGE MODELS AND ITS USE FOR UNBOUNDED-CONTEXT MEMORY

A PREPRINT

Hanzuo Liu^{†‡} Xuan Qi^{†*} Chunyu Liu^{†*} Haotian Zhong^{†*}
Yulong Wang[‡] Rayying[‡] Key[‡] Alex Lamb[†] Mingyu Gao[†]

[†]Tsinghua University [‡]Tencent

{lh24@mails., gaomy@}tsinghua.edu.cn

<https://github.com/liuhanzuo/COMem>

July 31, 2026

ABSTRACT

Transformer layers appear to play different roles: semantic information becomes accessible in lower and middle layers, whereas upper layers increasingly specialize representations for prediction. We turn this division of labor into **CoMem** (Comprehension Memory), which writes each context chunk only through an intermediate layer, retrieves a fixed number of cached residual tensor, and recomputes the query-conditioned upper layers over the resulting pack. For a fixed retrieval budget, model-side read compute and memory are independent of stored-context length. We evaluate a continued-trained Qwen3-8B base LM under a unified chat-template-free protocol. The backbone is frozen; the flagship trains only a rank-32 self-distillation LoRA on plain PG-19, and we report an adapter-free arm separately. CoMem reaches 97.05 on RULER and 38.27 on LoCoMo versus 34.59 for full-context KV-Direct; the dialogue-memory advantage survives conversation-cluster resampling and an independent judge. Results on additional long-context and long-document tasks expose both the benefits of bounded retrieval and its in-window compression tax. Controlled depth sweeps show that deeper caching lowers per-query recomputation but incurs a fidelity loss that self-distillation substantially repairs. In a separate adapter-free efficiency control on an NVIDIA H20 at 128k, CoMem uses 18.26 GB rather than 89.36 GB and achieves a $7.83\times$ prefill speedup. These results suggest that long-context memory can be organized along the layer axis, not only the token axis.

1 Introduction

Long-context language modeling incurs quadratic prefill cost from self-attention and linear memory growth from the key-value (KV) cache. (Vaswani et al., 2017; Kwon et al., 2023). It also inherits the backbone’s position range: Qwen3-8B (Yang et al., 2025) has a 40,960-token native window (the configured 131,072 limit requires inactive YaRN

scaling (Peng et al., 2023)), so we treat longer full-context runs only as unextended stress references.

Most memory-efficient methods evict or compress KV states (Xiao et al., 2024b; Zhang et al., 2023; Li et al., 2024; Cai et al., 2024; Liu et al., 2024a), retrieve relevant text or memory units (Lewis et al., 2020; Xiao et al., 2024a; Tang et al., 2024; Qiu et al., 2025b), or write into a trained memory (Wang et al., 2024, 2023b; Qiu et al., 2025a). Residual-stream reuse can reconstruct KV states without storing them independently (Qasim et al., 2026). CoMem combines bounded token selection with *depth-partitioned computation reuse*: it asks both which chunks to retrieve and how much of their computation can be reused. The premise is that lower and middle layers make semantic information accessible, whereas upper layers increasingly specialize it for the current query and prediction (Lad et al., 2024).

We validate this depth division with probes, causal truncation, and split-depth sweeps (§3), then turn it into **CoMem** (Figure 1). During WRITE, each chunk is processed only to depth j and its residual tensor is cached. During READ, a bounded set of relevant tensors is packed with the query and only layers $[j:L]$ are recomputed. The external store remains linear in stored context, but for fixed retrieval breadth the model-side online read does not. The flagship uses a rank-32 self-distillation LoRA (Hinton et al., 2015; Hu et al., 2022) on a frozen backbone; an adapter-free arm isolates the architecture.

Anonymous repository: <https://anonymous.open.science/r/COMem-Anonymous/>

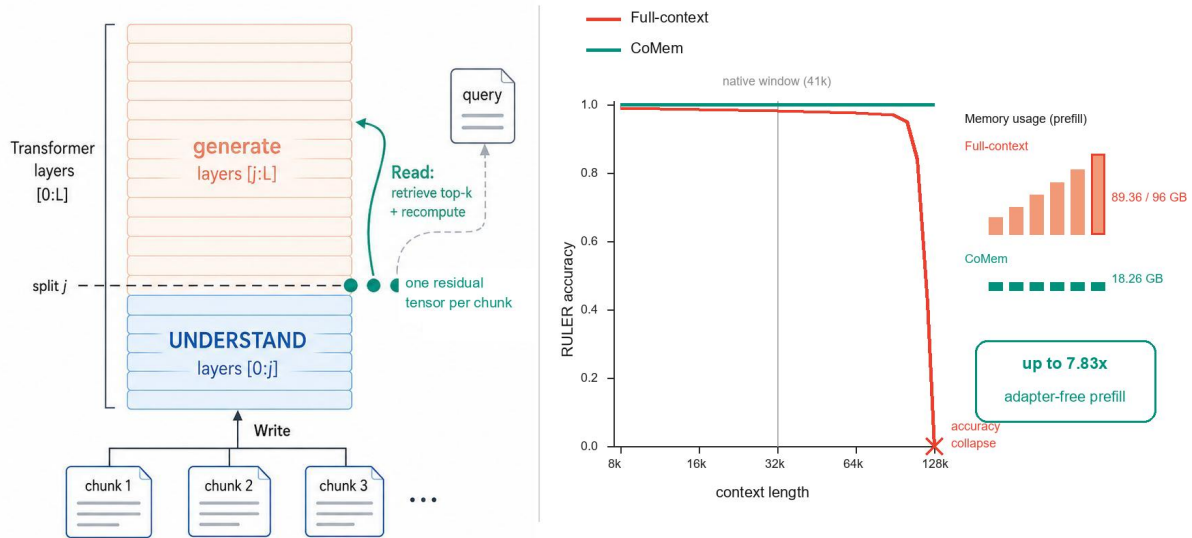


Figure 1: **CoMem in one picture.** Semantic information becomes usable by mid-depth, so CoMem caches the per-token residual tensor at depth j and recomputes the upper layers on a fixed retrieved pack. For fixed k , chunk size, and query length, the model-side online read working set is independent of stored-context length. The accuracy curve uses the distilled flagship, whereas the $7.83\times/18.26$ GB inset is a separate adapter-free efficiency control. CoMem remains accurate beyond the unextended backbone’s native window; we do not compare against a YaRN-extended full-context model.

Contributions.

- **Depth as a measurable reuse axis.** Semantic content becomes accessible near mid-depth, while the zero-shot readable boundary depends on model scale and can be shifted by lightweight adaptation.
- **A bounded depth-partitioned memory.** CoMem caches one residual state per token, exactly resumes the upper transformer, and combines bounded retrieval with full cross-chunk attention. On Qwen3-8B the cached state is $1/18$ the bytes of full bf16 KV per stored token.
- **Controlled quality and efficiency evidence.** Under a single chat-template-free Qwen3-8B evaluation protocol, CoMem reaches 97.05 on RULER and 38.27 on LoCoMo versus 34.59 for KV-Direct; the dialogue gain survives cluster resampling and an independent judge. A separate adapter-free H2O control at 128k uses 18.26 GB instead of 89.36 GB and achieves a $7.83\times$ full-write-inclusive prefill speedup. Ablations isolate retrieval, split depth, cross-chunk interaction, and distillation.

2 Related Work

We organize prior work along the same three axes that structure our design (§3): *what* to cache, *how deep* / how much to train, and *how* to read it back.

What to cache: full-depth KV vs. one intermediate residual tensor. The dominant approach to long context bounds the key–value (KV) cache along the *token* axis, retaining a compressed subset of *full-depth* KV. StreamingLLM (Xiao et al., 2024b) keeps a recent window plus attention-sink tokens; H2O (Zhang et al., 2023) and SnapKV (Li et al., 2024) evict all but heavy-hitter or predicted-attended positions; PyramidKV (Cai et al., 2024) and LCKV (Wu and Tu, 2024) reallocate or share the budget *across* layers; and MiniCache (Liu et al., 2024a) merges redundant KVs across neighboring layers. CompressKV (Lin et al., 2026) adds a retrieval-head eviction criterion. Closest to us, HCache (Gao et al., 2025) caches a mid-layer activation and recomputes the layers above it—but post-hoc, untrained, and without retrieval—and KV-Direct (Qasim et al., 2026) shows the KV cache is a deterministic projection of the residual stream and recomputes the *full-depth* KV exactly, yet keeps every token so its working set still grows with context. Most token-axis methods retain se-

lected full-depth KV, whereas HCache is the closest mid-depth-recompute design but lacks relevance selection. Following the depth division of labor (§3, *what to cache*) and the depth-semantics rationale of Lad et al. (2024), CoMem instead caches one depth- j residual tensor per chunk (one vector per token) and reconstructs the upper layers on demand.

How deep: fixed or trained memory vs. a tunable depth split. A second line fixes the memory by architecture or by training. MemoryLLM (Wang et al., 2024) maintains a fixed-size latent pool self-updated inside the transformer (we include its released Llama-3 checkpoint as an out-of-backbone reference), LongMem (Wang et al., 2023b) retrieves external states through a trained side network, and RecursiveSummarizing (Wang et al., 2023a) keeps a natural-language summary. YOCO (Sun et al., 2024) shares one global KV in a redesigned decoder-decoder, while CEPE (Yen et al., 2024) adds a parallel context encoder and cross-attention. Others induce compressibility through continued training: Activation Beacon (Zhang et al., 2024) learns to compress per-layer activations into beacon tokens, and KV-CAT (Gelberg et al., 2026) trains for cache compressibility on the token/slot axis. CoMem instead partitions on the *depth* axis and leaves the backbone frozen. Its inference-time memory operations require no parameter updates, while the evaluated flagship trains only a lightweight self-distilled LoRA; an adapter-free variant is reported separately. CoMem exposes the split j as a single RAG↔closed-book knob (§3, *how deep*).

How to read: retrieval-integrated memory vs. external recompute. Retrieval-augmented generation supplies selected text to a model (Lewis et al., 2020). Within long-context architectures, Landmark Attention (Mohtashami and Jaggi, 2023) selects blocks through learned landmark tokens; FoT/LongLLaMA (Tworkowski et al., 2023) uses an external contrastively trained (k, v) memory; and InfLLM (Xiao et al., 2024a) looks up token-relevant memory units inside attention. Quest (Tang et al., 2024) and RetrievalAttention (Liu et al., 2024b) instead retrieve query-relevant KV pages or vectors during attention. Serving systems such as CacheBlend (Yao et al., 2024) and RAGCache (Jin et al., 2024) precompute or reuse KV states for retrieved chunks. These methods differ in their stored unit and training require-

ments, but retrieval is coupled to token/KV memory or the attention path. CoMem keeps selection external: iterative BM25 (Robertson and Zaragoza, 2009) chooses chunk residual tensors, after which the upper layers are recomputed with full cross-chunk attention. A retrieved $j=0$ control separately measures selection without depth reuse.

3 Motivation

CoMem follows from one empirical property—transformer depth is not used uniformly—and three consequences for memory design. We keep only the evidence needed to motivate them here; controlled ablations are in §5 and Appendix A.1.

(I1) Cache an intermediate residual tensor.

Layer-wise probes place semantic accessibility near mid-depth, while causal truncation shows that a shallower prefix retains substantial downstream signal. Upper states are more query-conditioned (Lad et al., 2024), so a cached query-blind state should not be treated as final. CoMem therefore stores the complete per-token residual h_j and recomputes $[j:L]$ with the query present. On Qwen3-8B, h_j uses 1/18 the bytes of full bf16 KV per stored token.

(I2) Treat depth as a quality–cost knob. Recall remains high for shallow j and then falls as the cached state crosses a zero-shot readable boundary (Figure 2a). This boundary is distinct from the semantic *content* peak and moves deeper with model scale (Figure 2b). We expose j as a deployment knob and use self-distillation to move the operating point deeper; the flagship uses $j=12/36$. Exact $j=0$ equivalence and controlled depth/adaptation results appear in §5.

(I3) Read a bounded relevant pack. With the gold chunk present, an oracle selector reaches 100% on the reported RULER needle sweep, whereas recency and reader-attention selection fail as length grows (Figure 3). The observed bottleneck on this diagnostic is therefore selection, not cached-state fidelity. CoMem uses bounded relevance retrieval and full cross-chunk upper-layer attention; §5 tests retrieval breadth and chunk interaction directly.

4 Methodology

CoMem partitions a decoder along depth (Figure 4): Write caches one intermediate residual tensor per chunk; Select retrieves a bounded subset;

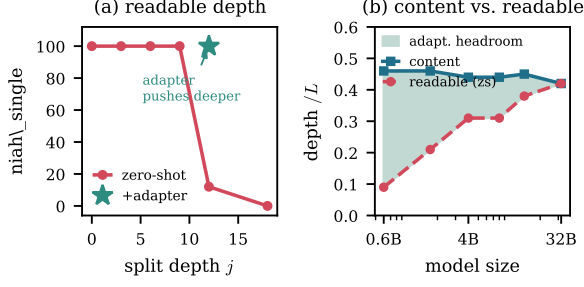


Figure 2: **The depth division of labor sets a bounded, tunable cache depth.** (a) On Qwen3-8B, zero-shot needle recall stays perfect while the split j is shallow and then collapses once the cache enters the query-conditioned band; a light self-distilled adapter pushes the readable depth deeper. Panel (a) is an independent trend sweep; Table 10 gives the controlled fixed- j cohort. (b) Across scale, the semantic *content* depth is near scale-invariant ($\approx 0.45L$) while the zero-shot *readable* depth deepens with model size; the shaded gap is the adaptation headroom suggested by the probe, and it vanishes by 32B. Full tables are in Appendix A.1.

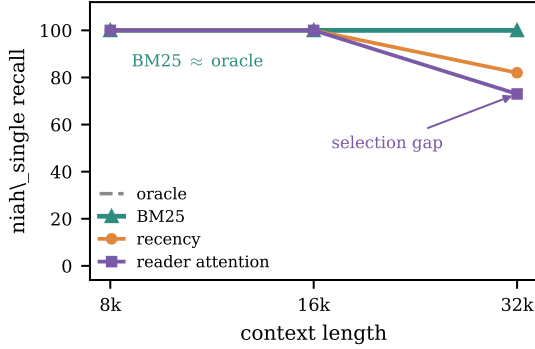


Figure 3: **Selection, not fidelity, is the observed long-range bottleneck.** On the chat-template-free RULER single-needle sweep, Oracle and BM25 remain at 100, while recency and reader attention fall at 32k. Full selector results are in Appendix Table 12.

Read recomputes the upper layers with the query present. The resumed computation is related to KV-Direct (Qasim et al., 2026), but CoMem combines it with shallow writes, a tunable split, and bounded external retrieval.

Notation and storage. Let the decoder have L layers, width d , per-layer KV width d_{kv} , split j , and chunk size c . CoMem stores $h_j \in \mathbb{R}^{c \times d}$. With a common dtype, its per-token storage relative to full KV is

$$\frac{|h_j|}{|KV|} = \frac{d}{2Ld_{kv}} = \frac{n_q}{2Ln_{kv}}, \quad (1)$$

where n_q and n_{kv} are the query- and KV-head counts. Qwen3-8B uses $d=4096$, $L=36$, $n_q=32$,

Algorithm 1 CoMem: Write and Read

Require: backbone layers $[0:L]$, split j , chunk size c , top- k

- 1: **Write** (streaming, once per context):
- 2: **for** each c -token chunk x in the context **do**
- 3: $h_j \leftarrow \text{layers}[0:j](x)$ {chunk-local RoPE}
- 4: store h_j keyed by the chunk text
- 5: **end for**
- 6: **Read** (per query q):
- 7: $S \leftarrow \text{LEXICAL-TOP-}k(q)$ {bounded top-12 flagship retrieval}
- 8: $\text{pack} \leftarrow [\text{sink}; \{h_j\}_{s \in S}; h_j(q)]$ {fresh contiguous RoPE}
- 9: $\text{logits} \leftarrow \text{layers}[j:L](\text{pack})$ {full cross-pack attention}
- 10: **return** next-token logits

and $n_{kv}=8$, yielding 1/18 (8 KB vs. 144 KB per token in bf16). At read time we assemble a *pack* whose length is fixed by k and c ,

$$|\text{pack}| = \text{sink} + k \cdot c + \text{query} \approx 6.7k, \quad (2)$$

for $k=12$, $c=512$, and—crucially—*independent of the context length*.

4.1 What to cache: the Write pass

The document is streamed in disjoint c -token chunks. Each chunk restarts rotary positions (Su et al., 2021) at zero, runs through $[0:j]$, and writes h_j to an external store keyed by its text. Chunk-local positions make cached states reusable regardless of their original document offset. The Write cost is linear in context length and executes only j layers. The hidden store and BM25 index also remain linear; only the model-side online Read is bounded.

4.2 How deep: the split j and self-distillation

The j knob. Split depth controls prepaid Write work versus per-query Read work. At $j=0$, CoMem is a selective full forward and matches the stock forward within $\max |\Delta \text{logit}| < 10^{-4}$ on the same pack; at $j=L$, no upper layer is replayed. Intermediate j trades cheaper Read for query-blind readout loss. Self-distillation enables the flagship operating point $j=12/36$. KV-Direct is the separate no-retrieval full-context arm.

Self-distillation. Following knowledge distillation and low-rank adaptation (Hinton et al., 2015; Hu et al., 2022), we train a student at $j=12$ to

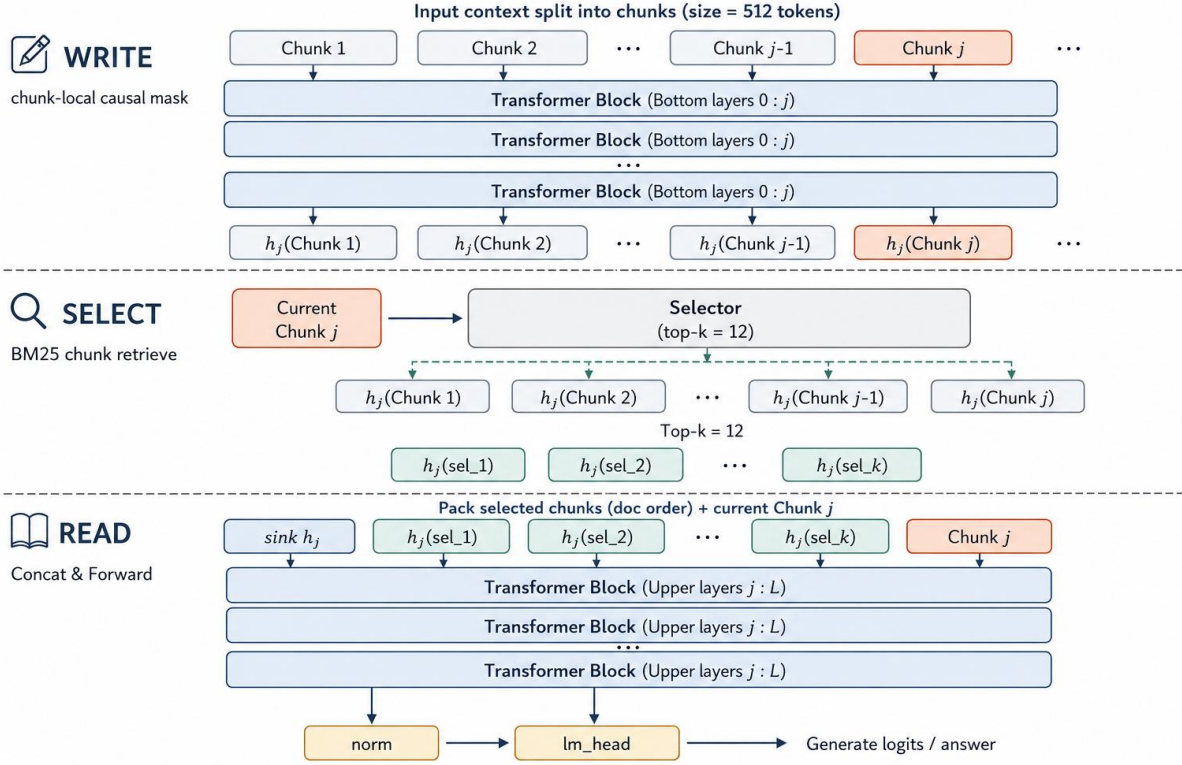


Figure 4: **CoMem pipeline.** WRITE encodes each 512-token chunk through $[0:j]$ and caches h_j . SELECT retrieves top- k relevant chunks. READ packs their hiddens with a sink and the query, then recomputes $[j:L]$ with full cross-chunk attention. The flagship reads approximately 6.5k tokens regardless of stored-context length.

match a frozen $j=0$ teacher on the teacher’s top-64 logit support. Let p and q be their renormalized distributions on that support. The actual objective is the bidirectional KL

$$\mathcal{L}_{\text{distill}} = 0.6 \text{KL}(p||q) + 0.4 \text{KL}(q||p), \quad (3)$$

with no auxiliary cross-entropy term. The canonical adapter trains rank-32 LoRA ($\alpha=64$, zero dropout) on all attention and MLP projections in layers 12–35 of a frozen Qwen3-8B. Training uses 4,000 steps of 8-GPU bf16 DDP (effective batch 8), AdamW with peak learning rate 8×10^{-5} , 100 warmup steps and cosine decay, on 2,048-token windows from the PG-19 training split (Rae et al., 2019). No synthetic long-context examples, task supervision, retrieval, or CE loss is used. This pushes the usable cache depth deeper and transfers without benchmark-specific supervision across the evaluated tasks (§5).

The backbone remains frozen. The $j=12$ flagship uses this LoRA, while a shallower adapter-free arm isolates the inference architecture. Designing compressible cache points during pretraining is complementary and outside our scope.

4.3 How to read: retrieval and recompute

The read pack. A selector returns top- k chunk indices. Their hiddens are ordered by document position and packed as $[\text{sink}; \text{selected } h_j; \text{query } h_j]$ with fresh, contiguous RoPE. Layers $[j:L]$ then run with full causal attention, allowing later chunks and the query to integrate earlier evidence. For fixed k and c , model-side Read FLOPs and KV working memory are independent of stored-context length; Table 6 tests the full-attention choice.

For generation, CoMem prefills the resumed upper-band KV once and maintains a query-local lower-band cache; it does not rerun the full pack per token. Decode work is therefore independent of stored-context length for a fixed pack and generated prefix. Table 3 verifies this path and reports timing.

Bounded lexical retrieval. BM25 (Robertson and Zaragoza, 2009) returns a bounded top- k set. The flagship uses $k=12$, hop width four, and three automatic frontier-expansion rounds with deduplication. Oracle, recency, reader-attention, and wider retrieval are analysis-only controls reported in §5

and Appendix A.1.

5 Experiments

Models and unified protocol. Unless noted, the backbone is the **Qwen3-8B base LM** (Yang et al., 2025) (36 layers), continued-trained without SFT or RL. All methods are evaluated under a unified plain-text prompting protocol, with chat templating disabled (`chat_template=False`). The backbone remains frozen; the flagship trains only our self-distilled rank-32 LoRA at split $j=12$ and uses chunk size 512, a BOS sink, and `iter_bm25`. All five benchmarks fix top-12, hop width four, and `rounds=0`, which automatically executes three retrieval rounds; the observed read is approximately 6.5k tokens. The native window is 40,960 tokens; 131,072 is an inactive YaRN (Peng et al., 2023) extrapolation ceiling.

Baselines and position range. KV-Direct (Qasim et al., 2026), the HCache-style reproduction (Gao et al., 2025), StreamingLLM (Xiao et al., 2024b), and InfLLM (Xiao et al., 2024a) use Qwen3-8B. KV-Direct runs with native RoPE and no YaRN; its results above 40,960 tokens are therefore an *unextended* full-context stress reference, not a substitute for a length-extended model. We claim continued usability where this unextended reference fails, not superiority to YaRN-enabled full context. For MemoryLLM (Wang et al., 2024), we use the official Llama-3-8B-chat backbone (Grattafiori et al., 2024) checkpoint released with its paper. It is an out-of-backbone reference. Because this chat-tuned checkpoint is evaluated without its native chat template, the resulting invocation is out of distribution and is interpreted only as a diagnostic reference.

Evaluation & metrics. Synthetic RULER (Hsieh et al., 2024) and BABILong (Kuratov et al., 2024) tables contain 100 examples per task-length cell. RULER uses official `string_match_all`; BABILong uses `compare_answers` with `TASK_LABELS` and never `re.search`. LongBench (Bai et al., 2023) and LoCoMo (Maharana et al., 2024) use `run_scoring`. LongEval follows the LongChat lines-retrieval protocol (Li et al., 2023). LoCoMo reports deterministic token F1 and substring accuracy as lexical diagnostics, with semantic Judge as the primary metric: GPT-4o scores categories 1–4, while category 5 uses the official local abstention rule. Complete RULER

grids passed the eight-shard completeness check, contained no empty predictions, and matched scores recomputed from disk. Appendix B gives aggregation and uncertainty details.

Hardware. Headline Qwen3-8B timing uses one 96 GB NVIDIA H20 (median of three runs, chunk size 512). At 128k, full-context prefill peaks at 89.36 GB and CoMem at 18.26 GB. A separate same-H20 split-depth sweep holds the retrieved pack fixed and is reported only within that cohort. Large-MoE details are in Appendix A.5.

5.1 Core benchmark overview

Table 1 compares five audited benchmarks under the unified evaluation protocol described above. CoMem scores 97.05 on RULER and 69.0 on LongEval versus 78.80/65.2 for the unextended KV-Direct reference, while matching its LongBench macro F1 (12.15 vs. 12.17). Frozen $j = 9$ is weak on multi-fact retrieval, and within-window BABILong exposes a qa1/qa2 compression tax, whereas the distilled model leads qa5. The advantage is therefore a bounded read, not uniform accuracy dominance; fidelity depends on selection, depth, and adaptation. Appendix A.2 gives full grids.

5.2 Equal-budget comparison

At the same ~ 6.5 k-token budget, CoMem stays near 100 on single needles and above 95 on variable tracking through 128k, while StreamingLLM falls $85 \rightarrow 2$ and $41.8 \rightarrow 0.8$ (Appendix Table 13). With read length approximately fixed, the gap shows that relevant rather than merely recent chunks are essential.

5.3 Long-dialogue memory

CoMem scores 38.27 versus 34.59 for KV-Direct on all 1,986 LoCoMo questions. The advantage is not an artifact of one scorer or treating correlated questions as independent: on 1,540 paired answerable items, the +4.81 gap has item-bootstrap 95% CI [2.34, 7.27]; a 10-conversation cluster bootstrap remains positive (8/10 conversations favor CoMem); and a stratified 200-item DeepSeek-V3 audit preserves the ordering ($\kappa = 0.626$ with GPT-4o). Retrieved $j = 0$ reaches 41.59, showing that retrieval contributes independently of depth reuse, while the same- j adapter control below measures how much fidelity is recovered. Appendix B details both uncertainty analyses and judge prompts.

Method	RULER [↑] 15-cell macro	LongEval [↑] 8k–128k	LongBench [↑] macro F1	BABILong [↑] 3-task macro	LoCoMo Judge [↑] full set	Avg. [↑] 5 benchmarks
CoMem + LoRA ($j=12$)	97.05	69.0	12.15	50.43	38.27	53.38
CoMem frozen ($j=9$)	59.41	3.2	10.63	39.19	29.15	28.32
KV-Direct	78.80	65.2	12.17	63.00	34.59	50.75
InfLLM	77.83	21.6	11.86	59.52	22.21	38.60
StreamingLLM	23.37	30.8	11.11	48.14	25.63	27.81
MemoryLLM [†]	16.55	13.6	9.01	30.00	16.11	17.05
HCache-style	3.73	0.0	9.20	33.71	8.11	10.95

Table 1: Cross-benchmark comparison under the unified plain-text evaluation protocol, with model-specific chat templating disabled. Avg. is the unweighted mean of the five displayed benchmark metrics, each reported on a 0–100 scale. Full-context Qwen3 results beyond the native 40,960-token window use native RoPE without YaRN and are therefore unextended stress references rather than length-extended upper bounds. The two CoMem rows separate the distilled and frozen configurations. [†]MemoryLLM uses an out-of-backbone Llama-3-8B-chat checkpoint and is included only as a diagnostic reference.

Method	Judge	Judge _{1:4}	F1	Substr. acc.
CoMem + LoRA ($j=12$)	38.27	48.64	9.15	23.36
CoMem frozen ($j=9$)	29.15	37.27	7.28	16.41
KV-Direct	34.59	43.83	9.02	22.36
StreamingLLM	25.63	31.04	7.67	13.75
InfLLM	22.21	26.43	7.39	13.34
MemoryLLM [†]	16.11	20.65	5.91	9.52
HCache-style	8.11	10.13	4.67	6.29

Table 2: LoCoMo comparison on all 1,986 chat-template-free examples. Judge_{1:4} is GPT-4o accuracy on the 1,540 answerable items; the full Judge column folds in local abstention scoring for 446 adversarial items. F1 and substring accuracy are deterministic lexical diagnostics; semantic Judge is the primary metric. All seven rows use the same scoring protocol. The distilled and frozen rows show the two deployment points; the same- j adapter effect is isolated in Table 8. On paired answerable items, distilled CoMem exceeds KV-Direct by 4.81 points (item-bootstrap 95% CI [2.34, 7.27]); a 10-conversation cluster bootstrap also excludes zero, and an independent DeepSeek-V3 audit preserves the ordering. [†]For MemoryLLM, we use the official chat-tuned Llama-3 checkpoint released with its paper; it is therefore an out-of-backbone reference.

5.4 End-to-end efficiency

Tables 3 and 4 separate three costs that are easily conflated. First, full-write-inclusive prefill becomes favorable after 8k and reaches $7.83\times$ at 128k while peak memory grows only from 17.33 to 18.26 GB; the full-context reference reaches 89.36 GB. Second, deployed resumed-band decode does not rerun the pack: it stays within 11% of Dense through 64k and becomes $1.07\times$ faster at 128k, with the correctness gate passing. Third, the separate single-column LoRA control shows that adaptation adds 0.25 GB and retains a $2.74\times$ prefill speedup. A fixed-pack depth sweep finds that

$j = 12$ lowers Read by 29% relative to $j = 0$; this diagnostic isolates upper-layer replay and is not mixed with cached decode. Appendix A.4 gives the measurement protocol.

5.5 What drives the result?

Retrieval versus depth reuse. The controlled depth sweep (Appendix Table 8) holds the retrieved pack fixed. The $j = 0$ row performs full-depth recomputation on it and therefore isolates selection from depth reuse. It reaches 41.59 on LoCoMo, compared with 34.59 for full-context KV-Direct, showing that bounded selection can improve natural dialogue memory even before layer reuse. Deeper frozen splits reduce quality (32.78/29.15/24.52 at $j = 6/9/12$) as query-blind states become harder to read, but a same-pack 128k sweep reduces Read from 1.01 to 0.72 s. Depth is therefore a quality–query-cost knob rather than a free accuracy gain.

Self-distillation repairs readout fidelity. At the same $j = 12$, the PG19-only LoRA restores 13.75 LoCoMo points and 22.2/9.0/8.4 BABILong qa1/qa2/qa5 points. The controlled RULER cohort similarly moves single-needle recall from 35/5/17 to 100/100/100 and multikey recall from 4/2/1 to 91/95/92 over 8k/16k/32k (Appendix Table 10). This supports adaptation of the split interface rather than benchmark-specific training: PG19 contains no retrieval or task supervision.

Selection breadth and chunk interaction. Oracle retrieval keeps the reported needle cells at 100, whereas recency and reader-attention selectors degrade with length (Appendix Table 12). More retrieval is not automatically better: the $\sim 17k$ four-hop variant is worse than the focused $\sim 6.5k$ flag-

(a) Full-write-inclusive prefill						(b) Resumed-band KV-cache decode					
Length	Full time	CoMem time	Speedup	Full mem.	CoMem mem.	Length	Full ms/tok	CoMem ms/tok	Speedup	Full mem.	CoMem mem.
8k	0.217 s	0.223 s	0.97×	19.90 GB	17.33 GB	8k	23.78	26.55	0.90×	19.7 GB	17.9 GB
16k	0.516 s	0.325 s	1.59×	24.53 GB	17.39 GB	16k	23.80	26.60	0.89×	23.0 GB	17.9 GB
32k	1.412 s	0.570 s	2.48×	33.79 GB	17.51 GB	32k	23.70	26.59	0.89×	29.6 GB	17.9 GB
64k	4.385 s	1.006 s	4.36×	52.31 GB	17.76 GB	64k	23.71	26.50	0.89×	42.8 GB	17.9 GB
128k	15.014 s	1.917 s	7.83×	89.36 GB	18.26 GB	128k	28.14	26.39	1.07×	69.1 GB	17.9 GB

Table 3: Qwen3-8B end-to-end efficiency ($j=12$, bf16, SDPA, H20-class platforms). **(a)** Adapter-free full-write-inclusive prefill: at 128k, CoMem is 7.83× faster and uses 18.26 GB versus 89.36 GB. **(b)** Resumed-band cached decode stays within 11% of Dense through 64k and is 1.07× faster at 128k; the correctness gate passes.

Arm	Prefill speedup	Peak mem.	Decode ms/tok
Dense	1.00×	89.39 GB	21.85
CoMem, LoRA off	3.23×	18.29 GB	23.78
CoMem, LoRA on	2.74×	18.54 GB	31.70

Table 4: Same-platform LoRA overhead at 128k. LoRA adds 0.25 GB peak memory and about 18% prefill time while preserving the fixed-read advantage. This is a separate paired-platform control; its ratios are not mixed with Table 3.

Retrieval configuration	8k	16k	32k	64k	128k
Auto 3-round, top-12 (~6.5k)	96.6	97.6	98.8	99.0	95.8
Hop-4 expansion (~17k)	99.0	95.6	89.8	89.8	87.4

Table 5: Retrieval breadth on RULER variable tracking ($n=100$) without a chat template. The first row is the flagship automatic three-round retrieval; the second widens the read to roughly 17k. They are distinct configurations and not an equal-budget causal comparison. The expansion point estimate is higher at 8k and lower from 16k onward, consistent with the larger read introducing distractor variable chains. More retrieval budget is not uniformly beneficial.

ship from 16k onward on variable tracking (Table 5), consistent with additional distractor chains. Once selected, chunks must interact: replacing full cross-chunk attention by block-diagonal reuse costs 36/62 points on 8k/16k multikey retrieval and 19/7 points on BABILong qa2 (Table 6). Together these controls justify bounded relevance selection, a tunable split, full cross-chunk recomputation, and lightweight interface adaptation as distinct parts of CoMem.

5.6 Additional tasks and generality

Appendix A.2 gives the remaining task details. Across Qwen3 0.6B–32B, the probed content depth stays near $0.45L$ while the zero-shot readable boundary moves deeper with scale; this is a depth

Task	Full attn.	Block-diag.	Δ
RULER niah_single	100 / 100	100 / 96	0 / +4
RULER niah_multikey	96 / 94	60 / 32	+36 / +62
BABILong qa2	36 / 20	17 / 13	+19 / +7
BABILong qa5	78 / 69	78 / 55	0 / +14

Table 6: Full causal cross-chunk attention versus block-diagonal reuse without a chat template (8k/16k). In both arms the final query attends to every retrieved chunk; the block-diagonal arm prevents cached chunks from attending across chunk boundaries. RULER uses $n=50$ and BABILong $n=100$, with the same top-12 iterative retrieval. Joint recomputation matters most for multi-fact disambiguation, while a single needle needs little interaction.

trend, not a claim of complete task transfer at every size. CoMem is also viable on two sparse MoE backbones. On Hy3, it keeps a ≈ 4.3 – 4.6 k-token read through 256k and reaches 100/98 single/multikey recall (Appendix A.5).

6 Conclusion

CoMem caches mid-layer chunk states and recomputes query-conditioned upper layers over a bounded, relevance-selected pack. Under the unified evaluation protocol, it reaches 97.05 on RULER, 69.0 on LongEval, 12.15 on LongBench, and 55.6/27.0/68.7 on BABILong. Its LoCoMo score is 38.27, compared with 34.59 for KV-Direct; the gap remains positive under a conversation-level cluster bootstrap and is preserved by an independent judge audit. Same-pack sweeps expose the central trade-off: deeper splits precompute more work and reduce query-time cost, but sacrifice frozen readout fidelity, while the $j = 12$ LoRA recovers 13.75 LoCoMo points without updating the backbone. CoMem therefore offers a quality–memory–compute trade-off rather than suggesting that depth cutting alone improves accuracy. Across Qwen3 model scales and two sparse MoE back-

bones, the results support depth as a complementary reuse axis for long-context memory.

7 Limitations

CoMem is a bounded-read efficiency method, not an accuracy state of the art for every in-window task. BABILong qa2 remains weak (27.0), and full context is a strong reference whenever it fits. Qwen3-8B is not position-extended in our experiments: results above its native 40,960-token window are unextended stress references and do not establish superiority over a YaRN-enabled full-context model. Retrieval breadth is task-dependent; retrieval latency on natural multi-hop workloads remains unmeasured. Full cross-chunk attention improves multi-fact accuracy but increases read compute.

Our evaluation is English-only and emphasizes retrieval-style QA rather than long-form generation, code, multilingual, or multimodal memory. We use greedy decoding, a 512-token flagship chunk size, and fixed top-12 BM25 retrieval; learned/dense retrievers, dynamic index updates, deletions, and continually changing memory streams are outside the present scope. Lexical retrieval may be less suitable when relevant passages share little surface form with the query.

Efficiency measurements use bf16/SDPA on H20-class hardware and a single-query harness. Kernel choice, batching, concurrent serving, storage I/O, and a networked external store can change absolute latency. Write is reusable and is most attractive when a context serves multiple queries; for one short query, preprocessing may not amortize. We do not measure index-build energy or total persistent-store capacity.

LoCoMo contains only 10 conversation clusters. We therefore supplement the item-level interval with conversation-cluster bootstrap and a stratified DeepSeek-V3 audit, but neither replaces evaluation on more conversations. The GPT-4o endpoint also lacks a dated snapshot. Synthetic cells use 100 examples, and not every ablation has multiple seeds or a paired test; chunk and attention differences are consequently reported as point estimates.

8 Ethical Considerations

CoMem changes the cost and storage organization of an underlying language model; it does not add a safety mechanism. It therefore inherits risks such as hallucination, biased generation, unsafe comple-

tion, and unintended disclosure of sensitive text selected from a memory store. Lower-cost access to long histories could also scale benign applications and deliberate misuse alike. Deployments should enforce access control on the persistent store, filter or redact sensitive sources before Write, respect deletion requests, restrict retrieval to authorized material, and retain application-appropriate output monitoring and human review. Although bounded reads reduce accelerator memory and per-query computation, training and large-scale evaluation still consume energy; we report the measured hardware and final training budget in Appendix A.3.

We collect no new human-subject data and recruit no annotators. Our experiments use previously released model, corpus, and benchmark artifacts; LoCoMo’s released conversations are generated by paired language-model agents rather than newly collected private conversations. The submitted software archive contains code only and excludes model weights, benchmark text, predictions, credentials, and API responses.

References

- Yushi Bai et al. 2023. LongBench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.
- Zefan Cai et al. 2024. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*.
- Shiwei Gao, Youmin Chen, and Jiwu Shu. 2025. Fast state restoration in LLM serving with HCache. In *Proceedings of the Twentieth European Conference on Computer Systems (EuroSys)*.
- Yoav Gelberg, Yam Eitan, Michael Bronstein, Yarin Gal, and Haggai Maron. 2026. Training transformers for KV cache compressibility. *arXiv preprint arXiv:2605.05971*.
- Aaron Grattafiori et al. 2024. [The Llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Cheng-Ping Hsieh et al. 2024. RULER: What’s the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

- Chao Jin et al. 2024. RAGCache: Efficient knowledge caching for retrieval-augmented generation. *arXiv preprint arXiv:2404.12457*.
- Yuri Kuratov et al. 2024. BABILong: Testing the limits of LLMs with long context reasoning-in-a-haystack. In *NeurIPS Datasets and Benchmarks*.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*.
- Vedang Lad, Wes Gurnee, and Max Tegmark. 2024. The remarkable robustness of LLMs: Stages of inference? *arXiv preprint arXiv:2406.19384*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuettler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33.
- Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. 2023. [How long can open-source LLMs truly promise on context length?](#) LM-SYS Org Blog.
- Yuhong Li et al. 2024. SnapKV: LLM knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*.
- Xiaolin Lin, Jingcun Wang, Olga Kondratieva, Yiyu Shi, Bing Li, and Grace Li Zhang. 2026. CompressKV: Semantic-retrieval-guided KV-cache compression for resource-efficient long-context LLM inference. *arXiv preprint arXiv:2606.24467*.
- Akide Liu, Jing Liu, Zizheng Pan, Yefei He, Gholamreza Haffari, and Bohan Zhuang. 2024a. MiniCache: KV cache compression in depth dimension for large language models. *arXiv preprint arXiv:2405.14366*.
- 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. 2024b. RetrievalAttention: Accelerating long-context LLM inference via vector retrieval. *arXiv preprint arXiv:2409.10516*.
- Adyasha Maharana et al. 2024. Evaluating very long-term conversational memory of LLM agents. *arXiv preprint arXiv:2402.17753*.
- Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint arXiv:2305.16300*.
- Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. YaRN: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.
- Kaleem Ullah Qasim, Jiashu Zhang, Muhammad Kafeel Shaheen, Razan Alharith, and Heying Zhang. 2026. The residual stream is all you need: On the redundancy of the KV cache in transformer inference. *arXiv preprint arXiv:2603.19664*.
- Jiahao Qiu, Xuan Qi, Hongru Wang, Xinzhe Juan, Yimin Wang, Zelin Zhao, Jiayi Geng, Jiacheng Guo, Peihang Li, Jingzhe Shi, et al. 2025a. Alita-g: Self-evolving generative agent for agent generation. *arXiv preprint arXiv:2510.23601*.
- Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, et al. 2025b. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *arXiv preprint arXiv:2505.20286*.
- Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.
- Stephen Robertson and Hugo Zaragoza. 2009. [The probabilistic relevance framework: BM25 and beyond](#). *Foundations and Trends in Information Retrieval*, 3(4):333–389.
- Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.
- Yutao Sun et al. 2024. You only cache once: Decoder-decoder architectures for language models. *arXiv preprint arXiv:2405.05254*.
- Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. 2024. Quest: Query-aware sparsity for efficient long-context LLM inference. In *International Conference on Machine Learning*.
- Szymon Tworkowski et al. 2023. Focused transformer: Contrastive training for context scaling. *arXiv preprint arXiv:2307.03170*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.
- Qingyue Wang et al. 2023a. Recursively summarizing enables long-term dialogue memory in large language models. *arXiv preprint arXiv:2308.15022*.
- Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. 2023b. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems*.

- Yu Wang et al. 2024. MemoryLLM: Towards self-updatable large language models. *arXiv preprint arXiv:2402.04624*.
- Haoyi Wu and Kewei Tu. 2024. Layer-condensed KV cache for efficient inference of large language models. *arXiv preprint arXiv:2405.10637*.
- Chaojun Xiao et al. 2024a. InfLLM: Training-free long-context extrapolation with an efficient context memory. *arXiv preprint arXiv:2402.04617*.
- Guangxuan Xiao et al. 2024b. Efficient streaming language models with attention sinks. In *ICLR*.
- An Yang et al. 2025. [Qwen3 technical report](#). *arXiv preprint arXiv:2505.09388*.
- Jiayi Yao et al. 2024. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. *arXiv preprint arXiv:2405.16444*.
- Howard Yen, Tianyu Gao, and Danqi Chen. 2024. Long-context language modeling with parallel context encoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.
- Peitian Zhang et al. 2024. Long context compression with activation beacon. *arXiv preprint arXiv:2401.03462*.
- Zhenyu Zhang et al. 2023. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *NeurIPS*.

Model	L	content- j ($/L$)	readout-crash ($/L$)	gap (Δ/L)
0.6B	28	13 (0.46)	~ 2.6 (0.09)	0.37
1.7B	28	13 (0.46)	~ 6 (0.21)	0.25
4B	36	16 (0.44)	~ 11 (0.31)	0.14
8B	36	16 (0.44)	~ 11 (0.31)	0.14
14B	40	18 (0.45)	~ 15 (0.38)	0.08
32B	64	27 (0.42)	>27 (> 0.42)	≈ 0

Table 7: **Depth partition: content vs. readable depth.** The *content- j* (the knee98 depth from a truncation-downstream linear probe) sits at $\approx 0.45L$ and is *near scale-invariant* over the evaluated 0.6B–32B range. The zero-shot *readout-crash* depth (where CoMem’s zero-shot single recall drops below 50% on RULER niah_single 16k) instead *deepens monotonically with scale*, from $0.09L$ (0.6B) to $> 0.42L$ (32B, no crash observed). Their difference is the **gap** the self-distilled adapter is intended to close: it is largest for the smallest model ($0.37L$ at 0.6B) and vanishes by 32B, where zero-shot readout already reaches the content peak. This trend predicts greater adaptation headroom at small scale; it is not a measured cross-scale adapter-gain ranking.

A Additional Results and Ablations

This appendix contains the detailed task tables, mechanism ablations, and cross-scale generality results summarized in the eight-page main paper. Appendix B separately documents aggregation, uncertainty, and the LoCoMo judge protocol.

A.1 Mechanism ablations

Split depth and self-distillation. Table 7 reports semantic *content* and adapter-off *readable* depth across Qwen3 scales. Table 8 gives the frozen $j = 0/6/9/12$ quality sweep and same- j distilled control; Table 9 pairs the same depths with a fixed-pack, same-H20 cost sweep. Table 10 provides a separate same- j RULER control. Finally, Table 11 applies the same adapter to a retrieval-free HCache path, showing that it is a portable mid-layer readout repair rather than a retrieval-specific component.

Selection and retrieval breadth. Table 12 compares Oracle, BM25, recency, and reader attention. Table 5 in the main paper contrasts the focused $\sim 6.5k$ configuration with a $\sim 17k$ four-hop expansion; these rows have different read budgets and are not a controlled per-hop causal estimate.

Read interaction and chunk size. Table 6 in the main paper compares full cross-chunk attention with block-diagonal reuse. Table 14 reports the

Configuration	Split j	LoRA	LoCoMo Judge	BABILong mean qa1 / qa2 / qa5
Retrieved full recompute	0	no	41.59	65.9 / 39.1 / 63.0
CoMem frozen	6	no	32.78	44.6 / 22.6 / 53.9
CoMem frozen	9	no	29.15	42.4 / 19.6 / 55.6
CoMem frozen	12	no	24.52	33.4 / 18.0 / 60.3
CoMem distilled	12	yes	38.27	55.6 / 27.0 / 68.7

Table 8: Controlled comparison of retrieval, split depth, and self-distillation using Qwen3-8B. All configurations follow the same plain-text prompting protocol, with model-specific chat templating disabled (`chat_template=False`), and retrieve the same top-12 pack. For the frozen configurations, readout fidelity decreases as the split moves deeper. At the same split $j=12$, the rank-32 LoRA improves the LoCoMo Judge score by 13.75 points and the BABILong qa1, qa2, and qa5 means by 22.2, 9.0, and 8.4 points, respectively. The $j=0$ configuration performs full-depth recomputation over the retrieved pack, thereby isolating the effect of relevance-based selection; it provides no lower-layer reuse and thus no depth-based reduction in query-time computation.

(a) Online Read latency						
Split j	Upper layers	8k	16k	32k	64k	128k
0	36	1.03	1.09	1.01	1.05	1.01
6	30	0.95	0.87	0.88	0.88	0.86
9	27	0.79	0.80	0.79	0.80	0.79
12	24	0.78	0.73	0.71	0.72	0.72

(b) Step, write, and memory at 128k			
Uncached step			
Split j	(s/token)	Write (s)	Peak (GB)
0	1.005	0.23	18.3
6	0.865	4.61	18.3
9	0.795	6.11	18.3
12	0.722	7.79	18.3

Table 9: Same-H20 split-depth cost sweep with no LoRA (median of three runs). Rows share an identical top-12 pack (about 6,657 tokens), chunk 512, and retrieval result. From $j = 0$ (retrieve then recompute all layers) to $j = 12$, 128k Read falls 29% and the uncached packed-step diagnostic 28%; the deeper split instead prepays reusable offline Write. The step diagnostic isolates upper-layer cost and is not the deployed KV-cache decode in Table 3.

chunk-size sweep; its entries are point estimates rather than a multi-seed stability ranking.

Task	Adapter	8k	16k	32k
niah_single_2	off	35	5	17
	on	100	100	100
niah_multikey_1	off	4	2	1
	on	91	95	92

Table 10: Separate controlled RULER self-distillation ablation on Qwen3-8B at the same split $j=12$ ($n=100$ per cell). Each on/off pair uses the same task, length, selector, and cohort, so the difference isolates the adapter. Table 8 provides the broader same- j LoCoMo/BABILong control used in the main argument.

Configuration	LoRA	LoCoMo Judge
HCache ($j=12$)	no	13.29
HCache ($j=12$)	yes	31.17

Table 11: Portable readout adaptation on a retrieval-free HCache path. The two configurations use the same compute node, code commit, evaluation harness, split, and plain-text prompting protocol, with model-specific chat templating disabled (`chat_template=False`). The distilled LoRA is the only changed component and improves the LoCoMo Judge score by 17.88 points. Because this within-harness control belongs to a different measurement cohort, its values should not be pooled with the canonical cross-node HCache result.

Task	Len	BM25	Recency	ReaderAttn	Oracle
niah_single	8k	100	100	100	100
	16k	100	100	100	100
	32k	100	82	73	100
niah_multikey	8k	99	98	97	100
	16k	99	88	90	100
	32k	99	54	60	100
var-track	8k	99.4	99.2	99.8	N/A
	16k	92.6	92.4	92.4	N/A
	32k	32.0	41.2	27.8	N/A

Table 12: Single-pass selector sweep on RULER ($n=100$) without a chat template; each entry is the peak over $k \in \{4, 8, 12, 16, 24\}$. Oracle retrieval keeps both needle tasks at 100, showing that their long-range gap is selection rather than readout loss. The variable-tracking rows belong to this specific selector-sweep configuration and should not be conflated with the focused top-12 flagship in Table 13. Oracle is undefined for a multi-chunk reference chain and is therefore not reported there.

Method	RULER task	8k	16k	32k	64k	128k
CoMem	niah_single	100.0	100.0	99.0	99.0	100.0
	var-track	96.6	97.6	98.8	99.0	95.8
StreamingLLM	niah_single	85.0	36.0	20.0	10.0	2.0
	var-track	41.8	2.2	0.6	0.2	0.8

Table 13: Equal-budget, chat-template-free RULER comparison ($n=100$ per cell). CoMem uses chunk 512, top-12, a BOS sink, and a measured read of about 6.5k tokens; StreamingLLM retains the same nominal token budget by recency. Relevance-based retrieval remains stable while recency truncation degrades with source length.

chunk	read tokens	uncached step (s/token)	8k	16k	32k	64k
128	1,665	0.19	91.0	90.0	81.0	85.0
256	3,329	0.37	80.0	90.0	90.0	94.0
512	6,657	0.72	89.0	95.0	97.0	94.0
1024	13,313	1.60	100.0	89.0	92.0	94.0

Table 14: Chunk-size trade-off: observed RULER `niah_multikey` accuracy ($n=100$) and same-harness packed-step cost without a chat template. Cost is approximately linear in read length. It is an uncached full-pack diagnostic, not the deployed KV-cache decode in Table 3. Accuracy differences are point estimates rather than a multi-seed stability claim.

Method	Task	8k	16k	32k	64k	128k
CoMem + LoRA	single	100	100	99	99	100
	multikey	95	94	97	91	93
	var-track	96.6	97.6	98.8	99.0	95.8
CoMem frozen ($j=9$)	single	96	99	99	99	96
	multikey	44	44	59	30	48
	var-track	45	38.8	36.4	31.2	25.8
KV-Direct	single	100	100	100	100	0
	multikey	100	100	98	88	0
	var-track	100	100	99.8	96.2	0
InfLLM	single	100	100	92	61	57
	multikey	100	79	65	45	20
	var-track	100	96.8	90.6	81.8	79.2
StreamingLLM	single	85	36	20	10	2
	multikey	83	34	18	13	4
	var-track	41.8	2.2	0.6	0.2	0.8
HCache-style	single	33	5	3	0	0
	multikey	8	4	0	0	0
	var-track	1.6	1.0	0.4	0	0
MemoryLLM [†]	single	29	40	37	21	21
	multikey	28	24	25	12	10
	var-track	0	1.2	0	0	0

Table 15: Full chat-template-free RULER comparison using official `string_match_all` ($n=100$ per cell). KV-Direct places the full context in the read with native RoPE and no YaRN; its 64k/128k cells are unextended stress references rather than a length-extended upper bound. CoMem keeps a bounded retrieved pack. The frozen arm remains strong on a single needle but degrades on multikey and tracking. InfLLM remains competitive in-window but degrades on long needle tasks, while CoMem stays above 91 throughout. [†]MemoryLLM uses its released Llama-3-8B-chat checkpoint.

A.2 Full chat-template-free benchmark comparisons

Tables 15–18 give the full audited baseline breakdowns for RULER, LongEval, BABILong, and LongBench. Every row uses the unified no-chat protocol; MemoryLLM remains an out-of-backbone Llama-3 reference. The all-method Lo-CoMo comparison is Table 2; judge details are in Appendix B.

Method	8k	16k	32k	64k	128k	Mean
CoMem + LoRA	69	75	64	67	70	69.0
CoMem frozen ($j=9$)	8	0	4	0	4	3.2
KV-Direct	100	96	92	38	0	65.2
StreamingLLM	86	34	18	10	6	30.8
InfLLM	60	30	12	4	2	21.6
MemoryLLM [†]	22	22	16	6	2	13.6
HCache-style	0	0	0	0	0	0.0

Table 16: LongEval register-content accuracy. Unmarked rows use the unified chat-template-free protocol. Means use the common 8k–128k support. KV-Direct uses native RoPE without YaRN, so its 64k/128k cells are unextended stress references and the five-length mean is not a length-extended upper bound. CoMem’s separate six-length headline is 72.83 after including its 4k score of 92. The frozen CoMem control used 48 generation tokens rather than 16; both budgets exceed the single-number answer length. [†]MemoryLLM uses a different chat-tuned backbone.

Method	Task	0k	1k	2k	4k	8k	16k	32k	Mean
CoMem + LoRA	qa1	98	80	68	68	46	17	12	55.6
	qa2	26	44	43	44	23	8	1	27.0
	qa5	68	76	76	75	68	60	58	68.7
CoMem frozen ($j=9$)	qa1	98	59	71	57	7	1	4	42.4
	qa2	53	17	28	26	8	4	1	19.6
	qa5	70	74	65	60	41	39	40	55.6
KV-Direct	qa1	98	84	80	74	80	72	63	78.7
	qa2	58	53	50	46	49	49	37	48.9
	qa5	71	73	62	59	65	42	58	61.4
InfLLM	qa1	97	84	81	72	69	52	34	69.9
	qa2	51	58	41	47	41	39	30	43.9
	qa5	69	68	63	66	73	60	55	64.9
StreamingLLM	qa1	97	83	80	71	23	27	12	56.1
	qa2	49	56	44	49	19	11	4	33.1
	qa5	68	65	68	65	39	47	34	55.1
MemoryLLM [†]	qa1	52	46	35	28	23	17	12	30.4
	qa2	37	29	17	18	21	17	11	21.4
	qa5	50	45	39	35	35	34	29	38.1
HCache-style	qa1	96	63	53	15	3	0	0	32.9
	qa2	57	15	35	17	2	0	0	18.0
	qa5	75	72	69	64	51	16	5	50.3

Table 17: BABILong accuracy ($n=100$ per task-length cell), evaluated with the official `compare_answers` scorer and `TASK_LABELS`. Within the native context window, full-context KV-Direct is substantially stronger on qa1 and qa2, exposing the compression tax, whereas distilled CoMem achieves the highest qa5 mean, ahead of InfLLM (68.7 vs. 64.9). All methods are evaluated under the unified plain-text prompting protocol, with model-specific chat templating disabled (`chat_template=False`). [†]MemoryLLM uses the official Llama-3-8B-chat checkpoint rather than the Qwen3-8B backbone. Because it is evaluated without its native chat template, this setting lies outside the checkpoint’s intended prompting distribution; its results are included only as an out-of-backbone diagnostic reference.

Method	NQA	Qasper	Hotpot	2Wiki	MultiF.	Musique	Macro
CoMem + LoRA	4.12	11.01	11.62	12.83	25.41	7.91	12.15
CoMem frozen ($j=9$)	4.63	11.23	9.41	10.71	22.05	5.72	10.63
KV-Direct	3.70	11.82	12.68	12.03	25.30	7.49	12.17
InfLLM	2.99	11.35	12.08	12.50	25.33	6.94	11.86
StreamingLLM	3.49	11.64	11.02	12.19	22.42	5.88	11.11
HCache-style	2.56	10.71	7.33	9.19	20.39	5.05	9.20
MemoryLLM [†]	3.13	8.46	8.76	10.27	17.43	5.98	9.01

Table 18: Official LongBench QA token F1 by dataset. All methods are evaluated under the same plain-text prompting protocol, with model-specific chat templating disabled (`chat_template=False`). CoMem with LoRA achieves a macro F1 of 12.15, closely matching the 12.17 obtained by full-context KV-Direct, suggesting that bounded compression introduces little additional loss relative to the full-context reference under this protocol. The frozen CoMem configuration reports adapter-free performance at $j=9$. NQA denotes NarrativeQA, and MultiF. denotes MultiFieldQA-en. [†]MemoryLLM uses the official Llama-3-8B-chat checkpoint rather than the Qwen3-8B backbone. Because it is evaluated without its native chat template, this setting lies outside the checkpoint’s intended prompting distribution; its results are included only as an out-of-backbone diagnostic reference.

A.3 Reproducibility details

Tables 19–21 record the configuration, training, and evaluation settings used by the Qwen3-8B flagship. The archived configuration files and evaluation scripts accompany the code release.

Artifacts, licenses, and data scope. The study evaluates English text only. It uses Qwen3-8B-Base (Apache-2.0), PG-19 (Rae et al., 2019), RULER (Hsieh et al., 2024), BABILong (Kurato et al., 2024), LongBench (Bai et al., 2023), LongEval/LongChat (Li et al., 2023), and LoCoMo (Maharana et al., 2024). RULER and LongChat code are Apache-2.0; the LongBench repository is MIT-licensed, while its component datasets retain their upstream terms; BABILong code is Apache-2.0 and incorporates BSD-licensed bAbI tasks; LoCoMo is CC BY-NC 4.0. PG-19 contains pre-1919 Project Gutenberg books, for which the dataset page does not assert one uniform corpus license, so we do not redistribute its text. We likewise redistribute no model weights, benchmark examples, predictions, or API responses. The anonymous code archive contains source, documentation, pinned environment requirements, and third-party notices; users obtain all external artifacts from their original providers and remain responsible for their terms. Table 21 reports the evaluated supports and sample counts; LoCoMo category denominators are in Appendix B.

Software environment. The archived environment specifies Python 3.10+, PyTorch 2.10, Transformers 5.5.4, PEFT 0.10+, Datasets 2.14+, bf16, and SDPA. Evaluation uses the benchmark-specific official scorers named in Table 21; the archive records all non-default flags and provides a CPU correctness gate.

Compute accounting. The final adapter run used one node with 8 H20 GPUs for about 22 minutes, or approximately 2.9 H20 GPU-hours. Tables 20 and 3 report model size, trainable parameters, hardware, and measured timing protocol. Total GPU-hours across preliminary probes, failed runs, baseline generation, and all ablations were not consistently logged; we report this omission rather than extrapolating an unreliable total.

Efficiency measurement. The reported chunk-512 headline uses a 96 GB H20-class platform with PyTorch 2.10, Transformers 5.5.4, bf16, and SDPA; the archived log records the device index

but not the GPU model, so the hardware label follows the experiment registry. It performs one unmeasured warmup and reports the median of three repetitions. Full-context prefill times the complete forward; CoMem times document Write, CPU retrieval, query Write, and Read through logits. At 128k, full-context prefill peaks at 89.36 GB while CoMem peaks at 18.26 GB. Peak memory is `max_memory_allocated`. The separate clean decode control uses 20 greedy steps and verifies token-level correctness.

The depth-Pareto cohort (Table 9) runs on one H20 with the same bf16/SDPA harness and reports medians of three measurements. It fixes the retrieved chunk IDs, top-12 pack (about 6,657 tokens), chunk size 512, and no LoRA across $j \in \{0, 6, 9, 12\}$. Its online Read and uncached packed-step columns isolate the cost of replaying 36/30/27/24 upper layers. The latter intentionally reruns the pack per step and must not be compared to the deployed resumed-band KV-cache decode below. Offline Write is measured separately because cached states are reused across future queries.

Layer	Setting	Value
Backbone	Architecture	Qwen3-8B base LM; $L=36$, $d=4096$, 32 query heads, 8 KV heads, bf16, SDPA.
Backbone	Positions	Native window 40,960; RoPE $\theta=10^6$; no active RoPE scaling or YaRN.
Write	Cached state	Split $j=12$; disjoint 512-token chunks; chunk-local positions restart at zero; one bf16 residual tensor h_j per token.
Read	Pack	[BOS; selected h_j ; $h_j(q)$] in document order, with fresh contiguous RoPE and full causal cross-chunk attention. The top-12 pack is 6,657 tokens and averages about 6.5k on RULER.
Retrieval	BM25	Token-ID documents and bare-question token IDs; $k_1=1.5$, $b=0.75$, Robertson IDF with +1 smoothing; no lowercasing, stemming, or stop-word removal.
Retrieval	Iteration	$rounds=0$ means automatic $\lceil k/h \rceil$ rounds, not one-shot. Each round uses newly selected chunks as the next frontier and deduplicates the selected set.
Retrieval	Width	All five benchmarks fix $k=12$, hop width $h=4$, and therefore three automatic rounds. Chunk size 512 and the BOS sink are also fixed.
Read	Mask	The pack is one causal sequence: the final query sees all chunks; later chunks may attend to earlier chunks. In the block-diagonal ablation, chunks see only themselves and the sink, while the final query still sees all chunks.
Generation	Decoding	Greedy decoding (no sampling, temperature, top- p , or beams); no chat template and no thinking mode.
Adapter-free arm	Deployment	Frozen backbone, no LoRA, $j=9$; all other memory and evaluation settings are held fixed.

Table 19: Flagship CoMem inference configuration. The retrieval and read configuration is frozen across all five benchmarks; only benchmark harness parameters differ.

Setting	Value
Adapter	Rank 32, $\alpha=64$, dropout 0; Q/K/V/O and gate/up/down projections in layers 12–35 only. The frozen backbone has 58.20M trainable LoRA parameters (0.71%).
Teacher and objective	The same Qwen3-8B with adapters disabled and $j=0$ serves as the frozen teacher. On the teacher top-64 support, $\mathcal{L} = 0.6 \text{KL}(p\ q) + 0.4 \text{KL}(q\ p)$ with implicit temperature 1 and no CE term. The loss is applied only to the query segment.
Data	Plain PG19 train text; no synthetic long-context examples, retrieval, task labels, or benchmark supervision. Training uses 2,048-token windows, formed as four 512-token chunks.
Optimization	AdamW, $\beta = (0.9, 0.95)$, weight decay 0, peak LR 8×10^{-5} , 100-step linear warmup, cosine decay to zero, gradient clipping 1.0, seed 42, bf16, no gradient checkpointing.
Schedule	4,000 steps on 8 GPUs, one sample per GPU and no gradient accumulation (global batch 8), totaling about 65.5M tokens.
Compute	One node with 8×NVIDIA H20 GPUs. Logged throughput is about 24.5 samples/s, implying roughly 22 minutes wall-clock; training peak memory was not recorded.

Table 20: Self-distillation and LoRA training configuration. The adapter name’s “4k” denotes 4,000 optimization steps; the training sequence length is 2,048 tokens.

Benchmark	Tasks and support	Generation	Scoring and aggregation
RULER	Single needle, multikey, and variable tracking; 8k–128k; 100 examples/cell; 8 shards; seed 42.	48 tokens; 60 for variable tracking.	Official case-insensitive <code>string_match_all</code> ; equal-weight macro over 15 cells.
LongEval	Lines retrieval; 4k–128k; 100 examples/length; 8 shards; seed 1234.	16 tokens for the flag-ship; 48 for frozen controls.	First numeric answer exact match; six-length macro (five-length 8k–128k macro for matched baseline tables).
LongBench	Six QA datasets; 1,150 examples total; 8 shards; seed 42.	32/64/128 tokens by dataset.	Official multi-reference token F1; equal-weight macro over six datasets.
BABILong	qa1/qa2/qa5; 0k–32k; 100 examples/cell; 4 shards; seed 42.	20 tokens.	Official <code>compare_answers</code> with <code>TASK_LABELS</code> .
LoCoMo	All 1,986 QA pairs from 10 conversations; 4 shards.	48 tokens for the main 8B comparison; 512 for the cross-scale audit.	GPT-4o judge on categories 1–4; official local abstention rule for category 5; full-set judge is the headline.

Table 21: Benchmark-level evaluation settings. Every main comparison uses `chat_template=False`. MemoryLLM uses a different chat-tuned backbone, so its no-chat row is interpreted only as an out-of-backbone reference.

A.4 Efficiency details

Table 3 in the main paper reports full-write-inclusive prefill, resumed-band decode, peak accelerator memory, and correctness; Table 4 separately reports same-platform LoRA overhead. Cross-platform timing ratios are not mixed.

A.5 Cross-scale and MoE generality

The Qwen3 depth sweep in Table 7 spans 0.6B, 1.7B, 4B, 8B, 14B, and 32B: content depth remains near $0.45L$, while the zero-shot readable boundary moves deeper with scale. Table 22 then evaluates a RULER-derived conservative integer split for each frozen backbone on all 1,986 LoCoMo questions. The no-chat score improves strongly overall (11.48→51.81) but is not strictly monotone. In particular, inspection of the 14B generations shows unusually long chain-of-thought trajectories without the chat template: reasoning consumes much of the generation budget before the final answer, plausibly explaining its dip below 8B. The chat template mitigates this behavior (28.80→39.58), while it slightly reduces 8B and 32B; we therefore treat the chat column as a generation-protocol sensitivity analysis rather than a second scaling curve. Table 23 adds Qwen3-30B-A3B, where Dense OOMs at 128k but CoMem runs at 63.0 GB and scores 100 on RULER single needle.

A.5.1 Hunyuan Hy3: an 80-layer sparse MoE

We additionally port CoMem to Tencent Hunyuan Hy3, an 80-layer sparse Mixture-of-Experts model (192 experts, top-8 routing plus one shared expert; ≈ 597 GB in bf16) with a 262,144-token native window. The backbone is sharded across eight GPUs; Write and Read semantics are unchanged.

Exact partition through MoE routing. Reconstructing a contiguous forward by writing `layers[0:j]` and resuming `layers[j:80]` gives $\max |\Delta \logit| = 0.0$ at $j \in \{0, 1, 40, 80\}$, including the Dense-to-MoE boundary. The resumed half re-executes every router and expert, so the partition preserves routing exactly on this self-test.

Split depth and self-distillation. A PG19 sweep places the best readout region near $j \approx 32$ –36. At $j = 32$, self-distillation reduces the 16k multiplicative LM tax from 1.496 to 1.078 and raises top-1 agreement from 0.773 to 0.871 (Table 24).

Bounded read through 256k. With the distilled adapter and BM25 top-8, the read stays at ≈ 4.3 –

Model	Safe j	j/L	No chat	Chat
Qwen3-0.6B	2	0.07	11.48	20.49
Qwen3-1.7B	3	0.11	17.42	26.44
Qwen3-4B	9	0.25	24.07	32.02
Qwen3-8B	9	0.25	32.38	31.52
Qwen3-14B	13	0.325	28.80	39.58
Qwen3-32B	27	0.42	51.81	48.89

Table 22: **Frozen CoMem transfers across the Qwen3 family on LoCoMo.** Each integer split is a conservative readout-safe point selected from RULER, independently of LoCoMo; it is not the 50% readout-crash boundary in Table 7. Values are full-set GPT-4o Judge accuracy (%) over all 1,986 questions. Both protocols use iterative BM25 (top-12, hop width 4), 512-token chunks, a BOS sink, greedy decoding, and a 512-token generation cap; the chat column is a template-sensitivity control. The 14B no-chat dip reflects unusually long chain-of-thought trajectories that delay the final answer; the chat template mitigates this behavior. Scores improve strongly with scale but are not strictly monotone.

Length	Prefill speedup	Dense mem	CoMem mem
8k	1.4×	63.7 GB	63.0 GB
32k	8.4×	71.3 GB	63.0 GB
128k	Dense OOM	OOM	63.0 GB

Table 23: CoMem on **Qwen3-30B-A3B** (48-layer sparse MoE; prefill speedup = Dense/CoMem prefill time). CoMem peak memory is context-independent (63.0 GB across 8k/32k/128k) while the full-context Dense reference grows and *OOMs* at 128k on an NVIDIA H20. At 128k CoMem still runs and scores RULER `niah_single` 100 (Dense OOM→CoMem 100); its prefill stays < 0.7 s at every length. The depth partition, retrieval, and constant read thus hold at sparse-MoE scale.

4.6k tokens from 16k through 256k. CoMem reaches 100/98 single/multikey recall at the 256k native ceiling with no OOM (Table 25).

	ctx 8k		ctx 16k	
	gap	top1	gap	top1
$j=32$				
zero-shot	1.378	0.823	1.496	0.773
distilled	1.071	0.895	1.078	0.871

Table 24: CoMem self-distillation on Hy3 at split $j=32$ (LoRA $r=32$, $\alpha=64$ on `layers[32:]`, teacher $j=0$ RAG recompute, 4000 steps of plain PG19 KL, no synthetic data). **gap** is the multiplicative LM tax (CoMem-readout perplexity / full-context perplexity; 1.0 is exact); **top1** is argmax agreement with the full forward, over matched 16-doc PG19 windows. The 16k LM tax falls from +49.6% to +7.8% and top1 rises $0.773 \rightarrow 0.871$, reproducing the 8B-backbone self-distillation effect on the 80-layer MoE.

Task	16k	32k	64k	128k	256k
niah_single_2	98	92	100	98	100
niah_multikey_1	100	94	100	100	98
read length (tok)	4.3k	4.6k	4.5k	4.4k	4.5k
context (tok)	16k	32k	65k	131k	261k

Table 25: CoMem on the Tencent Hunyuan **Hy3** backbone (`hy_v3`, 80-layer sparse MoE), RULER needle recall ($n=50$, official `string_match_all`), split $j=32$, BM25 top-8, self-distilled LoRA. The read length fed to the model stays ≈ 4.3 – 4.6 k tokens while the context grows $16\times$ ($16k \rightarrow 256k$); CoMem retains 100/98 recall at the backbone’s 256k native ceiling, with 0 out-of-memory failures. The context row reports the mean packed length of the RULER samples.

B Chat-Template-Free Statistical Verification

All statistics in this appendix were recomputed from the saved prediction shards on the shared result filesystem, without GPU reruns. The Qwen3-8B flagship uses the self-distilled $j=12$ adapter, chunk 512, top-12 retrieval, a BOS sink, and no chat template. The adapter artifact is `outputs/qcmem_distill_qwen_j12_r32_4k/final`; this internal path is reported only for reproducibility. The main RULER directory uses the `iter_bm25` implementation; the saved configuration records `rounds=0` and an observed read of approximately 6.5k tokens. Every reported RULER cell contains eight of eight shards, 100 examples, no empty prediction, and zero mismatch between the stored score and a fresh official-kernel recomputation. BABILong likewise contains 100 valid examples in every cell.

Task	8k	16k	32k	64k	128k
niah_single_2	100.0	100.0	99.0	99.0	100.0
niah_multikey_1	95.0	94.0	97.0	91.0	93.0
variable_tracking	96.6	97.6	98.8	99.0	95.8

Table 26: The 15 RULER cells entering the 97.05 headline. The score is their unweighted arithmetic mean: $1455.8/15 = 97.0533$. The separately evaluated 256k cells are excluded from this headline.

Task	8k	16k	32k	64k	128k	256k
niah_single_2	100.0	100.0	99.0	99.0	100.0	96.0
niah_multikey_1	95.0	94.0	97.0	91.0	93.0	91.0
variable_tracking	96.6	97.6	98.8	99.0	95.8	99.0

Table 27: CoMem RULER scaling without a chat template ($n=100$ per cell). The 97.05 headline averages only the 15 cells from 8k through 128k; 256k is a separately reported extension and is excluded from that headline.

B.1 RULER headline definition

Table 27 reports the separate 256k extension. Those three cells are excluded from the 97.05 headline to keep its predefined 8k–128k support.

B.2 LoCoMo judge protocol and denominator

Prediction shards for all methods and the adapter-free CoMem control deduplicate to all 1,986 LoCoMo items. GPT-4o judges the 1,540 answerable items in categories 1–4. Category 5 contains 446 adversarial abstention items and is scored locally: an answer is correct iff it is empty or matches the refusal rule. Thus the canonical full-set judge is over all 1,986 items, not only the API-judged subset.

Method	Cat. 1	Cat. 2	Cat. 3	Cat. 4	Cat. 5	Overall
CoMem + LoRA	26.95	19.00	30.21	69.32	2.47	38.27
CoMem frozen ($j=9$)	18.79	13.40	27.08	53.75	1.12	29.15
KV-Direct	24.11	18.69	25.00	62.19	2.69	34.59
StreamingLLM	22.70	11.21	23.96	42.21	6.95	25.63
InfLLM	18.44	14.33	25.00	33.89	7.62	22.21
MemoryLLM	15.60	8.72	15.63	27.47	0.45	16.11
HCache-style	6.74	2.49	9.38	14.27	1.12	8.11

Table 28: LoCoMo GPT-4o judge by category. Categories 1–4 contain 282/321/96/841 judged items; Category 5 contains 446 locally scored abstention items. Overall aggregates all 1,986 examples.

The judge uses the OpenAI-compatible endpoint <https://maas-openapi.wanjiedata.com/api/v1/chat/completions> with model name `gpt-4o`, `seed=1`, and no client-set temperature or `top-p`. The endpoint does not expose a dated model snapshot. Requests are retried

four times with exponential backoff; unparseable responses and API failures are conservatively scored wrong. The prompt presents the question, gold answer, and prediction; it accepts any semantic match (including equivalent date phrasing), rejects contradictions, omissions, empty answers, and refusals, and requires exactly CORRECT or WRONG. The verbatim template is released with the evaluation script. Gold alternatives are joined with “OR.” A reply must begin with CORRECT or WRONG; a fallback substring vote is used before defaulting to wrong.

On the common GPT-4o-judged items ($n=1540$), CoMem scores 48.64 and KV-Direct 43.83. A paired per-item bootstrap of their difference (10,000 resamples, seed 1234) gives +4.81 points with 95% CI [2.34, 7.27]. Because the questions are nested within only 10 conversations, this interval is not our sole inferential check. A paired conversation-cluster bootstrap, resampling the 10 conversations rather than individual questions, also yields a 95% interval entirely above zero; 8 of the 10 observed conversation-level differences favor CoMem. We report the latter as a dependence-aware robustness check rather than claiming that 1,540 questions are independent. The canonical full-set point scores are 38.27 and 34.59, respectively.

B.3 Independent-judge audit

We reran the binary prompt with DeepSeek-V3 on a stratified 200-item subset shared by CoMem and KV-Direct. Agreement with GPT-4o is 0.81 ($\kappa = 0.626$). Both judges preserve the ordering: GPT-4o scores 36.5/32.5 (+4.0) and DeepSeek-V3 56.0/49.0 (+7.0) for CoMem/KV-Direct. Absolute calibration differs, but the positive gap is judge-robust.

B.4 Uncertainty and aggregation checks

Per-method, 1,000-resample item-bootstrap intervals (seed 1234) are [36.20, 40.58] for CoMem’s full-set LoCoMo judge, [32.48, 36.71] for KV-Direct’s, [46.04, 50.98] and [41.23, 46.30] for their category-1–4 judge scores, and [8.60, 9.72] and [8.51, 9.61] for their F1 scores. These unpaired, item-level intervals are descriptive; the paired cluster analysis above is the appropriate dependence-aware comparison.

Selected 100-example cell intervals (1,000 resamples, seed 2024) are: variable tracking 64k, 99.0 [98.2, 99.8], and 128k, 95.8 [94.2, 97.4]; sin-

gle needle 32k, 99.0 [97, 100]; and multikey 64k, 91.0 [85, 96]. We omit legacy BABILong intervals computed from an earlier prediction cohort. LongEval scores 72.83 over 600 examples (95% CI [69.33, 76.33]).