ReContext: Recursive Evidence Replay as LLM Harness for Long-Context Reasoning

Yanjun Zhao*, Ruizhong Qiu*, Tianxin Wei*, Yuanchen Bei, Zhining Liu,
Lingjie Chen, Ismini Lourentzou, Hanghang Tong, Jingrui He†

University of Illinois Urbana-Champaign
{yanjunzh, jingrui}@illinois.edu

Abstract

Understanding and reasoning over long contexts has become a key requirement for deploying large language models (LLMs) in realistic applications. Although recent LLMs support increasingly long context windows, they often fail to use relevant evidence that is already present in the input, revealing a gap between context access and effective context utilization. In this work, we propose **Recursive Evidence Replay as LLM Harness for Long-Context Reasoning** (ReContext), a training-free inference method for improving long-context reasoning. ReContext uses model-internal relevance signals to construct a query-conditioned evidence pool and replays it before final generation while preserving the full original context. This recursive selection process separates evidence organization from answer generation without training, external memory, or context pruning. We also provide a theoretical analysis based on associative memory, which characterizes the context as a memory store, the question as a retrieval cue, attention as cue-trace association, and replay as trace reactivation. Experiments on eight long-context datasets with 128K context length show that ReContext consistently improves evidence utilization across Qwen3-4B, Qwen3-8B, and Llama3-8B, achieving the **best average rank** on all three backbones. Code is available at https://github.com/Yanjun-Zhao/ReContext.

1 Introduction

Long-context large language models (LLMs) can now place entire documents, multi-document collections, and extended dialogues inside a single prompt. However, longer context windows do not guarantee reliable long-context reasoning. A recurring failure mode is that the evidence needed to answer a question is already present in the input, but the model does not consistently use it during generation (Yen et al., 2025; Ye et al., 2026; Bei et al., 2026) This suggests that the bottleneck lies not only in *context access*, but also in *context harnessing*: we need a mechanism that dynamically manages long contexts during reasoning by continuously identifying, organizing, and updating the information most relevant to the current stage, thereby enabling more grounded and efficient long-context reasoning.

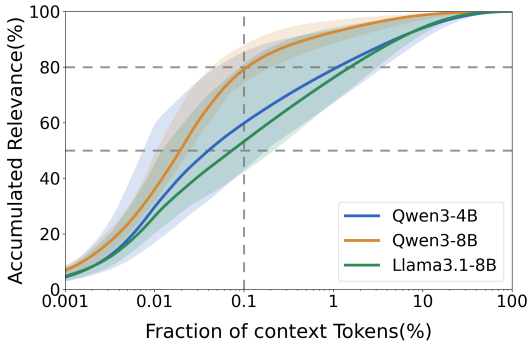

Figure 1: **Top 0.1% of context tokens already accounts for about 50% / 80% accumulated relevance score across three LLMs, corresponding to only 128 tokens in a 128K-token context.** This figure ranks all context tokens by their relevance scores with respect to the question and shows how much accumulated relevance score is covered by the top-ranked tokens. Each curve represents the mean trend over eight datasets, and the shaded region shows the variance across datasets.

Standard long-context prompting requires a model to read the context and answer the query in a single inference pass. As the context grows, the evidence must compete with increasing irrelevant information, making it harder for the model to ground its answers and leading to errors or hallucinations (Li et al., 2024; Liu et al., 2025). Recent efforts address this problem from different angles: attention intervention methods modify low-level model behavior (Li et al., 2024; Tang et al., 2024; Ye et al., 2026), making them invasive

*Equal contribution. †Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 26531–26548
October 24-29, 2026 ©2026 Association for Computational Linguistics

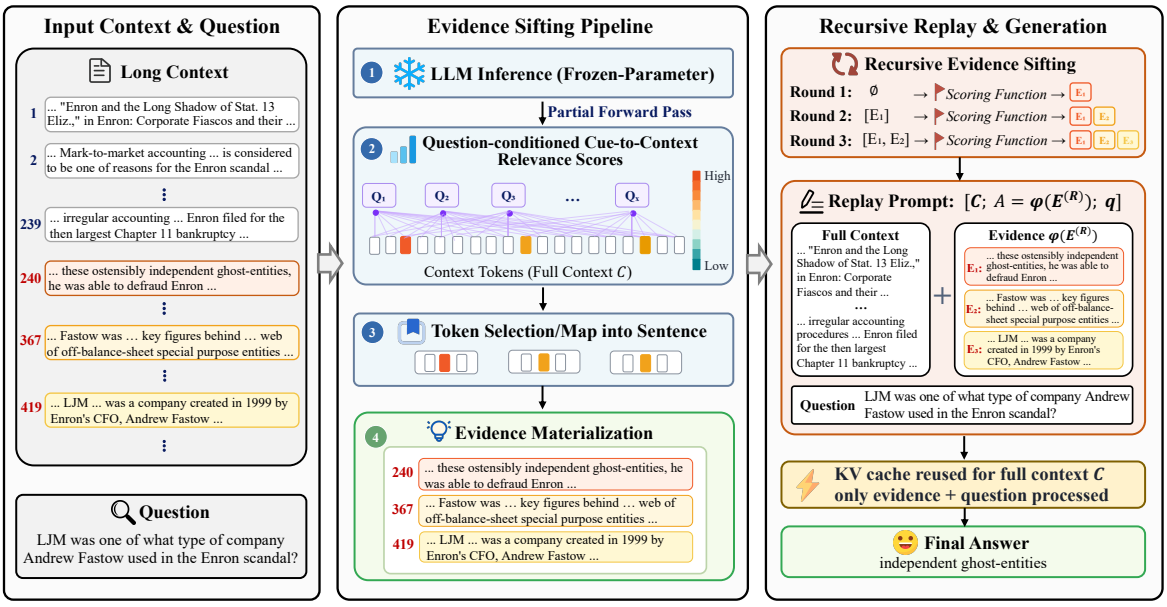

Figure 2: **Overview of ReContext.** ReContext identifies question-relevant evidence from a long context using internal LLM relevance signals, materializes selected tokens into grounded evidence spans, and recursively replays the resulting evidence before final generation while preserving access to the full context.

because they require changing the backbone forward or decoding logic. Retrieval and external-memory methods add retrieval systems (Lewis et al., 2021; Xu et al., 2025), while compression methods shorten the effective input (Jiang et al., 2023, 2024; Zhao et al., 2025c), both often rely on retrieved, compressed, or LLM-summarized evidence views, which may lose fine-grained details and become unstable on complex multi-hop tasks. This suggests that a context harness should not be limited to retrieving or reducing the input. Instead, while preserving full access to the original input, it should dynamically query and maintain a repository of supporting evidence according to the current reasoning stage, helping the model perform high-quality reasoning over long contexts.

We propose **Recursive Evidence Replay as LLM Harness for Long-Context Reasoning** (ReContext), a training-free inference method for long-context reasoning. Given a long context and a question, ReContext reads the original prompt, uses question-conditioned internal attention as candidate evidence proposals, materializes selected tokens as grounded text spans, and replays these spans before final answer generation. As shown in Figure 1, 128 tokens in a 128K-token context, already account for roughly 50–80% of accumulated question-conditioned relevance. ReContext turns these sparse signals into an evidence pool: across a small number of rounds, ReContext updates an ordered evidence pool by conditioning each new selection step on the original context, the question, and the evidence pool accumulated so far. This scaffold serves as a temporary external workspace in an iterative pipeline, which is why we view ReContext as LLM harness between full-context reading and final generation. The replayed scaffold changes the model state from which the next round's query-token attention scores are computed, allowing later rounds to surface evidence related to previously selected spans. The full context remains in the prompt; the evidence pool is used for emphasis, not exclusion. We use "recursive" in this limited inference-time sense: each evidence proposal depends on the evidence pool produced by previous rounds, rather than on an open-ended reasoning loop.

We further provide theoretical insights for ReContext from the perspective of associative memory. The long context can be viewed as a memory store, the question as a retrieval cue, attention as a prompt-internal proxy for cue-trace association, and replay as reactivation of selected traces near generation time. This view yields a monotonic-improvement proof showing that recursive evidence replay can move the hidden representation toward the answer embedding, and frames ReContext as query-evidence rebinding with internal relevance signals rather than a trained retriever or context pruning.

In summary, the contributions of our work are:

- We introduce **Recursive Evidence Replay as LLM Harness** (RECONTEXT), a training-free method that converts prompt-internal relevance signals into an explicit recursive evidence pool while preserving the full context.
- We provide an associative-memory explanation of RECONTEXT and a monotonic-improvement proof, formalizing evidence selection as cue-trace association and replay as trace reactivation for query-evidence rebinding.
- We evaluate RECONTEXT on eight 128K long-context datasets across Qwen3-4B, Qwen3-8B and Llama3-8B, where it achieves the best average rank on all three backbones and improves mean accuracy over Vanilla from 0.24 to 0.30, a **24.6% relative gain**.

2 Related Work

2.1 Long-Context Utilization and Evidence-Guided Reasoning

Recent work has significantly extended the context window of large language models (LLMs), enabling them to process long documents, multi-document inputs, code repositories, and extended dialogue histories. Existing methods improve long-context modeling through position extrapolation, efficient attention mechanisms, and long-context fine-tuning (Peng et al., 2023; Chen et al., 2024; Ding et al., 2023, 2024). Along with these modeling advances, a series of benchmarks have been proposed to evaluate long-context capabilities across diverse scenarios, including document question answering, multi-document reasoning, retrieval, summarization, code completion, and synthetic stress tests (Bai et al., 2024; Shaham et al., 2023; An et al., 2023; Zhang et al., 2024; Hsieh et al., 2024; Yen et al., 2025). These studies show that long-context evaluation should not only measure whether a model can accept long inputs, but also whether it can locate and use relevant information hidden in long contexts.

However, increasing the context window alone does not guarantee effective context utilization. Prior analysis has shown that LLMs are sensitive to the position of relevant evidence and may fail to use information when it appears in less favorable locations within the prompt (Liu et al., 2023a). To address this issue, retrieval-augmented generation retrieves relevant passages before generation (Lewis et al., 2021), while prompt and context compression methods reduce the input length by filtering, pruning, or compressing less informative content (Li et al., 2023; Jiang et al., 2023, 2024). Another line of work improves inference efficiency through KV cache compression or token eviction (Liu et al., 2023b; Zhang et al., 2023; Xiao et al., 2024).

2.2 Internal Attention Signals and Memory-Based Interpretation

Another related direction studies token-level importance inside LLMs. Prior work has observed that attention patterns in long-context inference are often highly structured: a small number of tokens may act as heavy hitters, attention sinks, or salient key positions that contribute disproportionately to future attention computation (Liu et al., 2023b; Zhang et al., 2023; Xiao et al., 2024). Based on this observation, recent methods select or retain important KV cache tokens to improve inference efficiency while preserving model performance (Li et al., 2024; Cai et al., 2025; Tang et al., 2024; Feng et al., 2025).

These studies demonstrate that model-internal signals can provide useful information about which context tokens are important. Our work is related to this line of research, but differs in objective. Rather than proposing a new token-importance estimator or KV-cache compression strategy, RECONTEXT uses existing question-to-context relevance signals to construct grounded candidate evidence spans, and studies how replaying these spans can improve long-context answer generation. The selected evidence is question-conditioned and chosen according to its relevance to the current question, rather than its general contribution to maintaining the cache. This use of relevance signals also admits an associative-memory interpretation: selected spans are treated as context traces associated with the query and replayed before generation. Classical and modern associative memory models retrieve stored patterns using partial or noisy cues, and recent studies have connected Transformer attention with memory retrieval mechanisms (Ramsauer et al., 2021; Krotov et al., 2025).

3 Method

3.1 Overview

Given a long context C and a question q, a standard long-context LLM generates directly from $[C; q]$. **Context Harness with Recursive Evidence Selec-**

Table 1: Main benchmark comparison across long-context tasks. RECONTEXT achieves the best average rank for all three backbones, showing consistent gains across datasets and model scales. Darker and lighter backgrounds indicate the best and second-best results within each backbone, respectively.

Model	Method	NQ 128K		TriviaQA 128K		HotpotQA 128K		PopQA 128K		NarrQA 128K		InfQA 128K		InfMC 128K	Clip. 128K	Avg Rank
		Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	Acc	
Qwen3-4B	Vanilla	0.02	0.21	0.04	0.24	0.00	0.10	0.00	0.11	0.02	0.17	0.09	0.21	0.51	0.38	4.39
	AttnSharp	0.02	0.21	0.02	0.23	0.00	0.09	0.00	0.10	0.03	0.17	0.10	0.23	0.47	0.42	4.25
	DySCO	0.02	0.21	0.10	0.30	0.03	0.13	0.00	0.10	0.01	0.17	0.11	0.23	0.50	0.44	4.00
	A-MEM	0.02	0.20	0.19	0.37	0.06	0.15	0.06	0.16	0.04	0.19	0.07	0.18	0.43	0.48	3.57
	DAC	0.02	0.18	0.21	0.38	0.07	0.17	0.01	0.09	0.05	0.20	0.07	0.19	0.43	0.24	3.79
	RECONTEXT	**0.08**	**0.25**	**0.30**	**0.45**	**0.08**	**0.19**	**0.07**	**0.19**	**0.07**	**0.21**	**0.12**	**0.24**	**0.55**	**0.52**	**1.00**
Qwen3-8B	Vanilla	0.06	0.26	0.53	0.66	0.18	0.31	0.18	0.32	0.19	0.33	0.22	0.36	**0.64**	0.30	3.96
	AttnSharp	0.06	0.26	0.43	0.59	0.18	0.31	0.13	0.29	0.18	0.34	0.23	0.37	0.60	0.32	4.50
	DySCO	0.09	0.30	0.51	0.64	0.19	0.31	0.22	0.35	0.18	0.33	0.23	0.36	0.63	**0.34**	3.25
	A-MEM	0.08	0.28	0.58	0.67	0.21	0.32	0.19	0.30	0.17	0.31	0.18	0.28	0.58	0.18	4.21
	DAC	0.12	0.29	0.65	0.72	**0.22**	**0.35**	0.15	0.30	0.14	0.28	0.15	0.26	**0.64**	0.20	3.61
	RECONTEXT	**0.13**	**0.33**	**0.68**	**0.75**	0.20	0.34	**0.23**	**0.36**	**0.21**	**0.35**	**0.25**	**0.39**	0.63	0.33	**1.46**
Llama3-8B	Vanilla	0.15	0.29	0.69	0.76	0.24	**0.39**	0.21	0.28	0.13	0.27	0.15	0.34	0.56	0.32	3.25
	AttnSharp	0.15	0.28	0.69	0.76	0.23	**0.39**	0.20	0.28	0.13	0.27	0.17	0.34	0.57	0.22	3.29
	DySCO	0.10	0.26	0.63	0.72	0.23	0.37	0.18	0.27	0.13	0.26	0.14	0.33	0.56	0.38	4.57
	A-MEM	0.16	**0.34**	0.67	0.75	0.24	0.34	0.17	0.23	0.15	0.29	0.19	0.33	0.54	0.32	3.43
	DAC	0.06	0.23	0.56	0.68	0.16	0.30	0.15	0.22	0.16	**0.30**	0.15	0.28	0.49	0.28	5.18
	RECONTEXT	**0.19**	0.31	**0.70**	**0.77**	**0.25**	**0.39**	**0.22**	**0.29**	**0.17**	0.29	**0.22**	**0.40**	**0.64**	**0.40**	**1.29**

tion (RECONTEXT) instead separates evidence organization from answer generation. It first reads the original prompt, extracts candidate evidence spans using question-conditioned internal relevance signals, replays these spans as an evidence pool, and then generates the final answer from the full context, the evidence pool, and the question. The method does not prune the prompt or directly modify attention logits during final decoding. The original context remains available throughout generation. The replayed scaffold emphasizes candidate evidence and makes evidence utilization explicit while keeping unselected context accessible.

RECONTEXT maintains an ordered evidence pool and updates it over a small fixed number of rounds. Each round computes relevance scores from a prompt that already contains the evidence pool accumulated in previous rounds.

3.2 Evidence Selection

Let M denote the backbone LLM. For a current prompt x, let $\mathcal{I}_x$ denote token positions in x, and let $\mathcal{I}_C \subseteq \mathcal{I}_x$ denote positions belonging to the original context C. Let $\mathcal{Q}(x) = (t_1, \ldots, t_L)$ be the last $L \leq w$ cue positions in the prompt suffix, with $w = 8$ in our main experiments. These suffix cues provide a query-conditioned readout, and in later rounds are conditioned on the replayed scaffold. During a read pass over x, we score prompt tokens by aggregating attention from these cue tokens over selected heads. For cue position t_u, define

$$a_i^{(u)} = \frac{1}{|\mathcal{H}|} \sum_{(l,h)\in\mathcal{H}} A_{t_u,i}^{(l,h)}, \qquad i \in \mathcal{I}_x, \tag{1}$$

where $\mathcal{H}$ is a selected set of layer-head pairs, and $A_{t_u,i}^{(l,h)}$ is the attention weight from cue token t_u to token i at layer-head pair (l, h). We accumulate cue-token evidence across $\mathcal{Q}(x)$ with exponential decay and normalization to obtain the final relevance score r_i. We then restrict candidates to the original context and select the top-K positions:

$$P = \mathrm{TopK}_{i\in\mathcal{I}_C}(r_i; K). \tag{2}$$

Here K is the evidence-token budget, and P contains the selected context positions. We treat these scores as an inexpensive prompt-internal proposal signal, which can identify candidate spans that may be useful for answer generation.

3.3 Evidence Materialization and Replay

Token-level proposals are often too fragmentary for answer generation. A selected token may identify an entity, date, or predicate, but the model usually needs the surrounding statement to use it reliably. Therefore, RECONTEXT maps selected token positions back to their containing sentences or local spans. Let $\mathcal{S}_C = (s_1, \ldots, s_N)$ be the ordered span decomposition of C, and let $\mathrm{pos}(s_n)$ denote the token positions covered by span s_n. The evidence

pool is the ordered subsequence of spans touched by selected tokens:

$$E = (s_n : \mathrm{pos}(s_n) \cap P \neq \emptyset)_{n=1}^{N}. \quad (3)$$

Each evidence unit is copied from the original prompt, so the evidence pool is grounded rather than freely generated.

For a single-pass replay, the prompt is

$$x^{+} = [C; \phi(E); q], \quad (4)$$

where $\phi(E)$ is the textual replay format. The answer is generated as

$$y \sim M(\cdot \mid x^{+}). \quad (5)$$

Replay places selected evidence near the question while keeping the full context available. In this sense, RECONTEXT selects for emphasis rather than exclusion: unselected context remains available during final generation.

3.4 Recursive Evidence Selection

For R rounds, RECONTEXT updates the evidence pool recursively. Let $E^{(0)} = \emptyset$ be the initial ordered evidence pool. At round $j \in \{1, \ldots, R\}$, the current prompt is

$$x^{(j-1)} = [C; \phi(E^{(j-1)}); q]. \quad (6)$$

The model reads $x^{(j-1)}$, so the replayed scaffold conditions the hidden states from which query-side prompt-suffix attention scores are computed. In the main setting, candidate positions are selected only from the original context token positions $\mathcal{I}_C$; the replayed scaffold conditions scoring but is not treated as a source of new copied spans. RECONTEXT then obtains relevance scores $r^{(j)}$, selects positions $P^{(j)}$, materializes spans from the original context, and appends only spans that are not already in the evidence pool. The span list proposed at round j is

$$\widehat{S}^{(j)} = (s_n : \mathrm{pos}(s_n) \cap P^{(j)} \neq \emptyset)_{n=1}^{N}. \quad (7)$$

The newly added spans are the ordered subsequence

$$\Delta E^{(j)} = \widehat{S}^{(j)} \setminus E^{(j-1)}, \quad (8)$$

and the evidence pool is updated by ordered concatenation:

$$E^{(j)} = E^{(j-1)} \cup \Delta E^{(j)}. \quad (9)$$

Here $\oplus$ denotes ordered concatenation; we avoid set union because the evidence pool is an ordered list. The final answer is generated from

$$x^{(R)} = [C; \phi(E^{(R)}); q]. \quad (10)$$

$$y \sim M(\cdot \mid x^{(R)}). \quad (11)$$

We call this process recursive because each round conditions on the evidence pool produced by previous rounds. In practice, R is small and fixed, so RECONTEXT remains a lightweight inference-time wrapper rather than an open-ended reasoning procedure.

3.5 Theoretical Analysis

In this subsection, we provide theoretical underpinning that our recursive evidence replay process in RECONTEXT can push the hidden embedding toward the answer. Our result is formally stated in Theorem 1 below.

Theorem 1 (monotonic improvement)**.** *Our theoretical setup is similar to prior works (Nichani et al., 2025; Olsson et al., 2022) and stated in Appendix E.1. Let $h^{(j)}$ denote the hidden embedding after the j-th evidence replay step, and let y denote the embedding of the answer. Then, for every step $j \geq 1$,*

$$\cos(h^{(j)}, y) > \cos(h^{(j-1)}, y), \quad (12)$$

where $\cos(\cdot, \cdot)$ denotes cosine similarity.

The proof is involved and thus deferred to Appendix E.2. Intuitively, our Theorem 1 shows that explicitly adding the evidence to the prompt can monotonically increase the similarity of the hidden embedding between the prediction and the answer.

4 Experiments

4.1 Datasets

We evaluate RECONTEXT on eight long-context benchmarks: Natural Questions (NQ), TriviaQA, HotpotQA, PopQA, NarrativeQA, InfBench QA, InfBench MC, and CLIPPER. For the first seven datasets, we adopt the 128K-context versions constructed by HELMET (Yen et al., 2025), while CLIPPER (Pham et al., 2025) evaluates evidence-grounded claim verification over long book contexts. These tasks cover factual question answering, multi-hop reasoning, narrative understanding, multiple-choice reasoning, and long-context claim verification. Detailed dataset descriptions are provided in Appendix B.1.

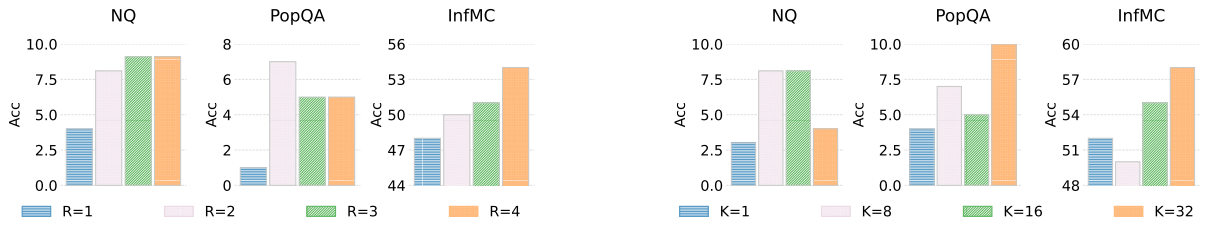

Figure 3: Visualization of the main ablation studies. Left: the effect of recursive evidence-selection rounds R. Right: the effect of the top-K evidence-token candidate budget.

We use the official metric associated with each task. For NQ, TriviaQA, HotpotQA, PopQA, NarrativeQA, and InfBench QA, we report answer accuracy (Acc) and token-level F1. For InfBench MC and CLIPPER, we report accuracy. Together, these tasks provide a broad testbed for long-context utilization.

4.2 Baselines

We compare RECONTEXT with the following:

- **Vanilla** directly generates from the full context and question.
- **AttnSharp** sharpens attention toward question-relevant context tokens, following attention-based long-context utilization methods (Tang et al., 2024; Ye et al., 2026).
- **DySCO** dynamically rescales decoding attention using retrieval-head signals (Ye et al., 2026).
- **A-MEM** stores and retrieves task-relevant context evidence with an external agentic memory module (Xu et al., 2025).
- **DAC** applies dynamic attention-aware prompt compression before generation (Zhao et al., 2025c).

In contrast, **RECONTEXT** preserves the full original context and replays a query-conditioned evidence pool before final generation.

4.3 Experimental Settings

We evaluate three backbone LLMs: Qwen3-4B, Qwen3-8B, and Llama3.1-8B. For each backbone, all methods use the same prompting format, decoding configuration, and context budget. Unless otherwise specified, all methods receive the same original long-context input and question, and the main results are obtained with thinking disabled. Detailed experimental settings are provided in Appendix C.

4.4 Main Results

The main comparison across eight datasets and three backbone models is summarized in Table 1. Task scores are reported as fractions in $[0, 1]$, while average rank is computed by ranking methods within each backbone on each reported metric column and averaging the resulting ranks; lower is better. RECONTEXT obtains the best average rank for all three backbones, with average ranks of 1.00 on Qwen3-4B, 1.46 on Qwen3-8B, and 1.29 on Llama3-8B. Averaging the eight Acc columns across all three backbones, RECONTEXT improves over Vanilla from 0.24 to 0.30, a **relative gain of 24.6%**. On Qwen3-4B, RECONTEXT achieves the best score on every reported metric, including improving NQ Acc from the strongest baseline score of 0.02 to 0.08.

On Qwen3-8B, RECONTEXT leads on most QA metrics and has the best average rank, with exceptions on HotpotQA, InfBench MC, and CLIPPER. On Llama3-8B, RECONTEXT obtains the best average rank and the highest accuracy score for every task, while NQ, HotpotQA, and NarrativeQA F1 are led by other baselines. Overall, these results indicate that explicit evidence replay improves aggregate long-context performance across model families, without implying dominance on every individual metric.

Under a shorter 64K context budget with thinking disabled, Table 3 shows that RECONTEXT stays within the top two on every reported metric. It ties for the best NQ Acc, obtains the best NQ F1, PopQA Acc, and InfBench MC accuracy, and ranks second on PopQA F1. Its macro-average over the five reported scores improves over Vanilla from 0.21 to 0.28, corresponding to a **35.0% relative gain** based on the reported scores. These results suggest that the observed improvements are not tied to a single context budget.

When thinking is enabled on Qwen3-4B, the robustness results in Table 2 show that RECON-

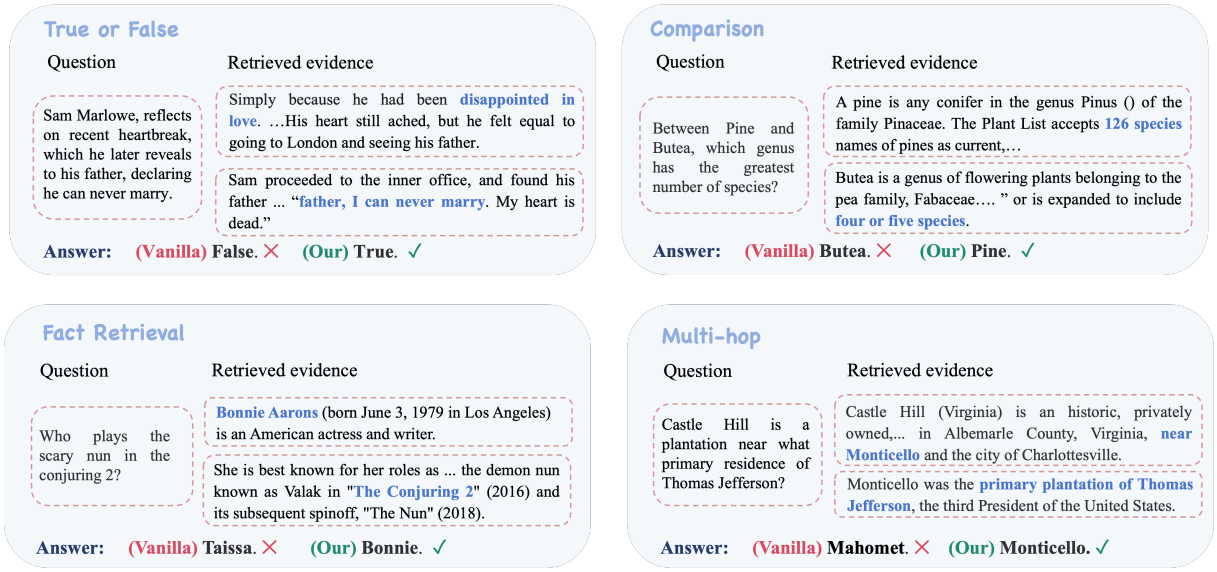

Figure 4: **Qualitative examples of RECONTEXT evidence replay.** RECONTEXT selects and replays query-relevant evidence spans (blue text) across diverse long-context reasoning tasks, enabling the model to ground its answer in the highlighted support and correct errors made by Vanilla generation.

Table 2: Robustness evaluation with thinking enabled. RECONTEXT remains strongest on NQ, PopQA and InfMC, indicating that evidence replay remains useful when the backbone performs explicit reasoning.

Method	NQ 128K		PopQA 128K		InfMC 128K
	Acc	F1	Acc	F1	Acc
Vanilla	0.08	0.24	0.14	0.25	0.69
AttnSharp	0.13	0.27	0.14	0.24	0.63
DAC	0.10	0.23	0.13	0.24	0.66
A-MEM	0.09	0.23	0.11	0.23	0.71
DySCO	0.14	0.29	0.17	**0.29**	0.67
RECONTEXT	**0.15**	**0.30**	**0.18**	0.28	**0.72**

Table 3: Robustness evaluation under a shorter context budget. RECONTEXT stays within the top two on every metric, suggesting that the gains are not tied to the main context length.

Method	NQ 64K		PopQA 64K		InfMC 64K
	Acc	F1	Acc	F1	Acc
Vanilla	0.07	0.24	0.04	0.20	0.48
AttnSharp	0.07	0.24	0.03	0.20	0.44
DAC	0.09	0.23	0.17	**0.32**	0.53
A-MEM	0.06	0.20	0.06	0.19	0.51
DySCO	**0.11**	0.26	0.07	0.23	0.46
RECONTEXT	**0.11**	**0.26**	**0.18**	0.30	**0.54**

TEXT achieves the best NQ Acc and F1, the best PopQA Acc, and the best InfBench MC accuracy. Its macro-average over the five reported scores improves over Vanilla from 28.0 to 32.6, a **relative gain of 16.7%**. PopQA F1 is an exception, where DySCO obtains the highest score. These results suggest that Recursive Evidence Selection remains useful even when the backbone model is allowed to produce intermediate reasoning.

4.5 Ablation Studies

We analyze the effects of recursive selection rounds and candidate-token budget in Tables 6 and 7, with selected accuracy trends visualized in Figure 3.

The number of recursive evidence-selection rounds R controls how many times the evidence pool is expanded before final replay. Moving from one round to two rounds improves all reported metrics, raising the macro-average from 0.17 to 0.22. Larger values provide further gains for NQ and InfBench MC, with the best NQ Acc at $R = 3$ and $R = 4$, the best NQ F1 and InfBench MC accuracy at $R = 4$, and the best PopQA scores at $R = 2$. This suggests that additional selection can improve performance on some tasks, but the best recursion depth is task-dependent rather than uniformly larger.

The top-K token candidate budget determines how many high-scoring token positions can seed evidence span materialization in each selection round. With $R = 2$ fixed, Table 7 shows that NQ Acc ties at $K = 8$ and $K = 16$, while NQ F1 peaks at $K = 8$; by contrast, PopQA and InfBench MC achieve their strongest scores at $K = 32$. The

Table 4: Ablation on evidence-token source. Selecting evidence from the original context consistently outperforms selecting from the full replay prompt.

Source	NQ 128K		PopQA 128K		InfMC 128K
	Acc	F1	Acc	F1	Acc
Full prompt	0.04	0.23	0.02	0.14	0.52
Context	**0.08**	**0.25**	**0.07**	**0.19**	**0.54**

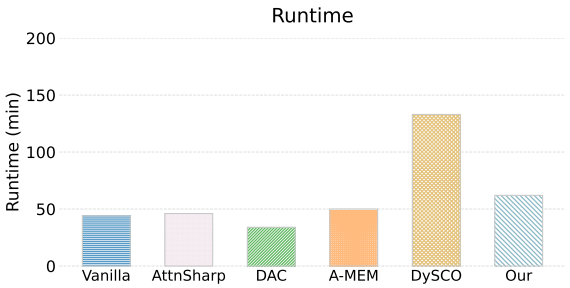

Figure 5: Runtime comparison on CLIPPER using Llama3-8B at 128K context length.

macro-average rises from 0.19 at $K = 1$ to 0.23 at $K = 32$, but the task-level pattern is not monotonic: larger candidate sets can expose more candidate spans, while smaller budgets can be cleaner for NQ.

Finally, the token-source ablation in Table 4 compares selecting evidence tokens from the original context only versus from the full replay prompt. Context-only selection consistently outperforms full-prompt selection on NQ, PopQA, and InfBench MC, improving the macro-average from 0.19 to 0.23. The largest gains appear on PopQA Acc, which increases from 0.02 to 0.07, and NQ Acc, which increases from 0.04 to 0.08. This supports the main setting: the replayed scaffold conditions the model state used for scoring, but copied evidence spans are selected from the original context rather than allocating the evidence budget over the entire prompt.

4.6 Qualitative Results

We further compare the methods through qualitative case analysis on retrieval-heavy examples, as shown in Figure 4. Vanilla receives the full context, but relevant spans may become weakly bound at generation time. AttnSharp and DySCO strengthen attention toward relevant tokens, but the selected evidence remains latent inside the decoding process rather than being exposed as text. A-MEM and DAC create shorter evidence views through memory retrieval or compression, which can help when the preprocessing step keeps the right span but can also weaken the answer if supporting sentences are omitted. By contrast, RECONTEXT copies model-selected evidence sentences into an explicit evidence pool and replays them near the question while leaving the original context intact. In the inspected cases, the evidence pool appears to encourage answers grounded in a compact set of supporting sentences rather than in a diffuse long prompt.

4.7 Time and Space Efficiency

As shown in Figure 5, RECONTEXT introduces the evidence-selection and replay stage, making it slightly slower than the vanilla baseline. However, it remains substantially faster than DySCO, which changes the backbone forward or decoding logic. In terms of GPU memory consumption, RECONTEXT inserts fewer than 128 additional evidence tokens, resulting in only minimal memory overhead. Consequently, its overall memory consumption is similar to both Vanilla. Detailed measurements are provided in Appendix B.3.

5 Conclusion

We present RECONTEXT, a training-free method that turns model-internal relevance signals into an explicit evidence scaffold for long-context reasoning. RECONTEXT recursively selects candidate evidence and replays the organized scaffold before answer generation while preserving the original context, thereby separating evidence organization from answer generation. Our associative-memory analysis provides a simple interpretation of this process as cue-conditioned trace reactivation, and our experiments across eight 128K-context datasets and three backbones show consistent improvements over strong long-context baselines. These findings suggest that long-context inference can be improved not only by extending context windows or compressing inputs, but also by better organizing the evidence already available in the prompt.

Acknowledgement

This work is supported by Agriculture and Food Research Initiative (AFRI) grant no. 2020-67021-32799/project accession no.1024178 from the USDA National Institute of Food and Agriculture and NSF (2433308). The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the

funding agencies or the government.

Limitations

RECONTEXT requires access to model-internal relevance signals, which limits its direct use with closed-source APIs that do not expose attention or similar scoring information. The method also adds a read-and-replay stage, so inference latency is higher than direct full-context decoding, although it remains training-free and does not maintain a persistent external memory.

References

Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *Preprint*, arXiv:2307.11088.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. Longbench: A bilingual, multitask benchmark for long context understanding. *Preprint*, arXiv:2308.14508.

Yuanchen Bei, Tianxin Wei, Xuying Ning, Yanjun Zhao, Zhining Liu, Xiao Lin, Yada Zhu, Hendrik Hamann, Jingrui He, and Hanghang Tong. 2026. Mem-gallery: Benchmarking multimodal long-term conversational memory for mllm agents. *Preprint*, arXiv:2601.03515.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, pages 2206–2240.

Trenton Bricken and Cengiz Pehlevan. 2021. Attention approximates sparse distributed memory. In *Advances in Neural Information Processing Systems*, volume 34.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, and Wen Xiao. 2025. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *Preprint*, arXiv:2406.02069.

Chao Chen, Tian Zhou, Yanjun Zhao, Hui Liu, Liang Sun, and Rong Jin. 2025. Does vector quantization fail in spatio-temporal forecasting? exploring a differentiable sparse soft-vector quantization approach. *Preprint*, arXiv:2312.03406.

Lingjie Chen, Ruizhong Qiu, Yuyu Fan, Yanjun Zhao, and Hanghang Tong. 2026. Trims: Trajectory-ranked instruction masked supervision for diffusion language models. *Preprint*, arXiv:2604.00666.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2024. Longlora: Efficient fine-tuning of long-context large language models. *Preprint*, arXiv:2309.12307.

Sizhe Dang, Yangyang Guo, Yanjun Zhao, Haishan Ye, Xiaodong Zheng, Guang Dai, and Ivor Tsang. 2025. Fzoo: Fast zeroth-order optimizer for fine-tuning large language models towards adam-scale speed. *Preprint*, arXiv:2506.09034.

Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. 2023. Longnet: Scaling transformers to 1,000,000,000 tokens. *Preprint*, arXiv:2307.02486.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens. *Preprint*, arXiv:2402.13753.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S. Kevin Zhou. 2025. Ada-kv: Optimizing kv cache eviction by adaptive budget allocation for efficient llm inference. *Preprint*, arXiv:2407.11550.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495. Association for Computational Linguistics.

Yangyang Guo, Yanjun Zhao, Sizhe Dang, Tian Zhou, Liang Sun, and Yi Qian. 2024. Less is more: Embracing sparsity and interpolation with esiformer for time series forecasting. *Preprint*, arXiv:2410.05726.

J. J. Hopfield. 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558.

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

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. Llmlingua: Compressing prompts for accelerated inference of large language models. *Preprint*, arXiv:2310.05736.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *Preprint*, arXiv:2310.06839.

Dmitry Krotov, Benjamin Hoover, Parikshit Ram, and Bao Pham. 2025. Modern methods in associative memory. *Preprint*, arXiv:2507.06211.

Dmitry Krotov and John J. Hopfield. 2016. Dense associative memory for pattern recognition. In *Advances in Neural Information Processing Systems*, volume 29, pages 1172–1180.

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. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Yucheng Li, Bo Dong, Chenghua Lin, and Frank Guerin. 2023. Compressing context to enhance inference efficiency of large language models. *Preprint*, arXiv:2310.06201.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024. Snapkv: Llm knows what you are looking for before generation. *Preprint*, arXiv:2404.14469.

Zihao Li, Kaifeng Jin, Yuanchen Bei, Jiaru Zou, Avaneesh Kumar, Xuying Ning, Yanjun Zhao, Mengting Ai, Baoyu Jing, Hanghang Tong, and Jingrui He. 2026. Harnessing generalist agents for contextualized time series. *Preprint*, arXiv:2606.05404.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023a. Lost in the middle: How language models use long contexts. *Preprint*, arXiv:2307.03172.

Zhining Liu, Rana Ali Amjad, Ravinarayana Adkathimar, Tianxin Wei, and Hanghang Tong. 2025. Selfelicit: Your language model secretly knows where is the relevant evidence. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9153–9173. Association for Computational Linguistics.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. 2023b. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. *Preprint*, arXiv:2305.17118.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409. Association for Computational Linguistics.

Eshaan Nichani, Jason Lee, and Alberto Bietti. 2025. Understanding factual recall in transformers via associative memories. In *International Conference on Learning Representations*, volume 2025, pages 4166–4207.

Xuying Ning, Katherine Tieu, Dongqi Fu, Tianxin Wei, Zihao Li, Yuanchen Bei, Jiaru Zou, Mengting Ai, Zhining Liu, Ting-Wei Li, Lingjie Chen, Yanjun Zhao, Ke Yang, Bingxuan Li, Cheng Qian, Gaotang Li, Xiao Lin, Zhichen Zeng, Ruizhong Qiu, and 23 others. 2026. Code as agent harness. *Preprint*, arXiv:2605.18747.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *Preprint*, arXiv:2209.11895.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.

Chau Minh Pham, Yapei Chang, and Mohit Iyyer. 2025. Clipper: Compression enables long-context synthetic data generation. *Preprint*, arXiv:2502.14854.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. 2021. Hopfield networks is all you need. *Preprint*, arXiv:2008.02217.

Tao Ren, Jinyang Jiang, Hui Yang, Wan Tian, Minhao Zou, Guanghao Li, Zishi Zhang, Qinghao Wang, Shentao Qin, Yanjun Zhao, Rui Tao, Hui Shao, and Yijie Peng. 2025. Riskpo: Risk-based policy optimization via verifiable reward for llm post-training. *Preprint*, arXiv:2510.00911.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *Preprint*, arXiv:2305.14196.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, volume 28, pages 2440–2448.

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

Tianxin Wei, Ting-Wei Li, Zhining Liu, Xuying Ning, Ze Yang, Jiaru Zou, Zhichen Zeng, Ruizhong Qiu, Xiao Lin, Dongqi Fu, Zihao Li, Mengting Ai, Duo Zhou, Wenxuan Bao, Yunzhe Li, Gaotang Li,

Cheng Qian, Yu Wang, Xiangru Tang, and 10 others. 2026a. Agentic reasoning for large language models. *Preprint*, arXiv:2601.12538.

Tianxin Wei, Zhan Shi, Minhua Lin, Bing He, Zewen Liu, Yisi Sang, Yuanchen Bei, Xuying Ning, Jiaru Zou, Ting-Wei Li, Xiao Lin, Yanjun Zhao, Chi Wang, Benoit Dumoulin, Dakuo Wang, Jingrui He, and Hanqing Lu. 2026b. Evo-harness: Context-to-harness skill compilation for self-evolving agents. *Preprint*, arXiv:2608.15071.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *International Conference on Learning Representations*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. *Preprint*, arXiv:2309.17453.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-mem: Agentic memory for llm agents. *Preprint*, arXiv:2502.12110.

Xi Ye, Wuwei Zhang, Fangcong Yin, Howard Yen, and Danqi Chen. 2026. Dysco: Dynamic attention-scaling decoding for long-context language models. *Preprint*, arXiv:2602.22175.

Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. 2025. Helmet: How to evaluate long-context language models effectively and thoroughly. *Preprint*, arXiv:2410.02694.

Qi Yu, Zhichen Zeng, Katherine Tieu, Xiyuan Yang, Ruizhong Qiu, Yuchen Yan, Lihui Liu, Yanjun Zhao, Lingjie Chen, Jingrui He, and Hanghang Tong. 2026. From inference to adaptation: A unified optimal transport view of vision language model. *Preprint*, arXiv:2608.18339.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. ∞bench: Extending long context evaluation beyond 100k tokens. *Preprint*, arXiv:2402.13718.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. 2023. H_2o: Heavy-hitter oracle for efficient generative inference of large language models. *Preprint*, arXiv:2306.14048.

Kai Zhao, Yanjun Zhao, Jiaming Song, Shien He, Lusheng Zhang, Qiang Zhang, and Tianjiao Li. 2025a. Saber: Switchable and balanced training for efficient llm reasoning. *Preprint*, arXiv:2508.10026.

Yanjun Zhao, Sizhe Dang, Haishan Ye, Guang Dai, Yi Qian, and Ivor W. Tsang. 2025b. Second-order fine-tuning without pain for llms:a hessian informed zeroth-order optimizer. *Preprint*, arXiv:2402.15173.

Yanjun Zhao, Ziqing‘ Ma, Tian Zhou, Mengni Ye, Liang Sun, and Yi Qian. 2023. Gcformer: An efficient solution for accurate and scalable long-term multivariate time series forecasting. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM ’23, page 3464–3473, New York, NY, USA. Association for Computing Machinery.

Yanjun Zhao, Tianxin Wei, Jiaru Zou, Xuying Ning, Yuanchen Bei, Lingjie Chen, Simmi Rana, Wendy H. Yang, Hanghang Tong, and Jingrui He. 2026. Papermind: Benchmarking agentic reasoning and critique over scientific papers in multimodal llms. *Preprint*, arXiv:2604.21304.

Yanjun Zhao, Tian Zhou, Chao Chen, Liang Sun, Yi Qian, and Rong Jin. 2024. Sparse-vq transformer: An ffn-free framework with vector quantization for enhanced time series forecasting. *Preprint*, arXiv:2402.05830.

Yi Zhao, Zuchao Li, Hai Zhao, Baoyuan Qi, and Guoming Liu. 2025c. Dac: A dynamic attention-aware approach for task-agnostic prompt compression. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19395–19407.

A Extended Related Work

Long-context modeling and evaluation. Recent long-context research has substantially expanded the input length that LLMs can process through efficient attention, position extrapolation, and long-context fine-tuning (Beltagy et al., 2020; Zaheer et al., 2020; Peng et al., 2023; Chen et al., 2024; Ding et al., 2023, 2024; Zhao et al., 2023; Ren et al., 2025; Ning et al., 2026; Guo et al., 2024; Wei et al., 2026a,a; Li et al., 2026). In parallel, long-context benchmarks have shifted evaluation beyond input length alone, covering document understanding, multi-document reasoning, retrieval, summarization, code, and synthetic stress tests (Bai et al., 2024; Shaham et al., 2023; An et al., 2023; Zhang et al., 2024; Hsieh et al., 2024; Yen et al., 2025; Zhao et al., 2026). These studies clarify an important distinction between context access and context utilization: a model may accept a long prompt while still failing to locate or use the evidence needed for a particular question. This gap is also reflected in analyses of position sensitivity, where relevant information can be underused when it appears in less favorable locations (Liu et al., 2023a). RECONTEXT addresses this utilization problem without changing the backbone model or extending its context window. It keeps the original context available, but adds a query-conditioned evidence scaffold that makes candidate supporting spans more accessible near generation time.

Retrieval, external memory, and prompt compression. Retrieval-augmented generation and retrieval-enhanced language modeling improve grounding by retrieving relevant passages or chunks before generation (Lewis et al., 2021; Borgeaud et al., 2022), while recent agentic memory systems maintain and query external memory structures across interactions (Xu et al., 2025). Prompt and context compression methods take a different route: they reduce the amount of text passed to the model by filtering, pruning, or compressing less informative content (Li et al., 2023; Jiang et al., 2023, 2024; Zhao et al., 2025c,a; Wei et al., 2026b; Yu et al., 2026; Chen et al., 2026). These approaches can reduce distraction and computation, but they also introduce an additional selection or compression step that may omit useful evidence. RECONTEXT is complementary to this line of work. It does not build a persistent memory, train a retriever, or replace the original long context with a shortened version. Instead, it uses the model's own prompt-internal signals to copy candidate evidence spans into an explicit scaffold, while leaving the full context in the prompt as a fallback source of information.

Attention-based inference and KV-cache methods. Several inference-time methods use internal attention patterns to improve long-context efficiency or utilization. Query-aware sparsity and dynamic attention-scaling methods use attention-derived signals to select relevant tokens or adjust attention behavior during decoding (Tang et al., 2024; Ye et al., 2026). A related line of KV-cache methods observes that long-context attention often concentrates on heavy hitters, attention sinks, or salient key positions, and uses this structure for cache retention or eviction (Liu et al., 2023b; Zhang et al., 2023; Xiao et al., 2024; Li et al., 2024; Cai et al., 2025; Feng et al., 2025; Zhao et al., 2025b; Dang et al., 2025). RECONTEXT differs in both objective and mechanism. It does not directly rescale attention logits during final decoding and does not optimize a cache budget. More importantly, it treats attention only as an inexpensive proposal signal rather than as a faithful explanation of model behavior. The selected spans are materialized as readable text, so the final generation is conditioned on an explicit evidence scaffold rather than only on latent attention intervention.

Associative-memory interpretation. The behavior of RECONTEXT can also be viewed through an associative-memory lens. Classical associative-memory models describe content-addressable retrieval from stored patterns, while dense and modern Hopfield-style formulations connect such retrieval with higher-capacity memory and attention-like updates (Hopfield, 1982; Krotov and Hopfield, 2016; Ramsauer et al., 2021; Krotov et al., 2025; Zhao et al., 2024; Chen et al., 2025). Related neural memory architectures, including Memory Networks and key-value memory networks, frame reasoning as query-conditioned access to stored representations or facts (Weston et al., 2015; Sukhbaatar et al., 2015; Miller et al., 2016). Transformer-specific analyses further connect attention or feed-forward components with associative and key-value memory views (Bricken and Pehlevan, 2021; Geva et al., 2021). In our setting, the long context acts as a collection of memory traces, the question serves as a retrieval cue, and attention provides a prompt-internal proxy for cue–trace association rather than a faithful explanation of

the model's decision process. Evidence sifting selects candidate traces, evidence materialization turns them back into grounded text spans, and replay reactivates these spans near answer generation. Under this view, recursive evidence sifting is a lightweight way to repeatedly re-query the same context under a scaffold-conditioned state, improving query–evidence rebinding without adding an external memory module, training a retriever, or removing the original context.

B Experiment Details

B.1 Dataset Details

The first seven datasets in our evaluation come from the HELMET benchmark (Yen et al., 2025), and CLIPPER is evaluated as an additional long-context claim-verification benchmark (Pham et al., 2025).

- **NQ** evaluates open-domain factual question answering, where the model must locate short answer evidence in a long input.
- **TriviaQA** also tests factual question answering, but with trivia-style questions that often require matching paraphrased clues to supporting evidence.
- **HotpotQA** focuses on multi-hop question answering and requires combining evidence from multiple pieces of context.
- **PopQA** probes entity-centric factual knowledge and is sensitive to whether relevant evidence about less prominent entities is correctly used.
- **NarrativeQA** evaluates narrative understanding over long stories, requiring the model to connect events, characters, and plot information.
- **InfBench QA** contains free-form question answering over very long inputs and stresses evidence retrieval from extended contexts.
- **InfBench MC** uses a multiple-choice format over long inputs, testing whether the model can select the option best supported by the context.
- **CLIPPER** evaluates claim verification over book-length contexts with evidence-grounded synthetic claims.

B.2 GPU Resources

We conduct experiments on NVIDIA A100 and NVIDIA H200 GPU servers. The H200 runs are executed on an ARM64 (`aarch64`) system architecture. All methods compared under the same setting use the same backbone, context budget, prompting format, and decoding configuration on the corresponding hardware.

Table 5: Runtime usage on CLIPPER using Llama3-8B at 128K context length.

Method	Runtime
Vanilla	44 min
AttnSharp	46 min
DAC	34 min
A-MEM	50 min
DySCO	2h 13min
RECONTEXT	62 min

B.3 Detailed Efficiency Analysis

We report wall-clock runtime on CLIPPER using Llama3-8B at 128K context length with thinking disabled. Vanilla full-context decoding takes 44 minutes, AttnSharp takes 46 minutes, DAC takes 34 minutes in total, A-MEM takes 50 minutes, and DySCO requires 2 hours and 13 minutes. The best-performing variant of RECONTEXT takes 62 minutes. The additional runtime of RECONTEXT mainly comes from evidence sifting and replay, which add computation beyond direct full-context decoding; nevertheless, RECONTEXT remains substantially faster than DySCO, which changes the backbone forward or decoding logic. Its GPU memory stays at the same level as Vanilla and DySCO because the replay scaffold adds fewer than 128 evidence tokens.

B.4 RECONTEXT Implementation Details

The RECONTEXT implementation is activated by setting `-decoding_method Our`. In this mode, the evaluation driver calls the sentence-replay path in `rescale_generate`: the model receives the original prompt, computes candidate evidence proposals from attention, reconstructs a replay prompt, and then generates from the replay prompt. The same custom model classes also support attention-rescaling baselines, but RECONTEXT uses the selected tokens to construct replayed evidence rather than to directly rescale attention logits during final decoding.

Prompt segmentation. For each dataset, the implementation identifies the boundary between the long-context portion and the question or answer-format suffix. Let L_C be the number of tokens before this boundary. The original prompt is split into context tokens $x_{1:L_C}$ and question-side tokens $x_{L_C+1:T}$. The default replay prompt is formed as context, replayed evidence, and question-side tokens. The code also contains alternative replay positions for ablations, but the multi-round setting uses the before-question replay form.

Attention readout. The readout uses the final w prompt tokens as cue tokens, where w is controlled by `context_warmup_steps` and is set to 8 in our main runs. For each cue token, attention weights are averaged over a fixed set of selected layer-head pairs. Scores are accumulated across cue positions with exponential decay:

$$r^{(t)} = \text{Normalize}\left(a^{(t)} + \lambda r^{(t-1)}\right), \qquad (13)$$

where $a^{(t)}$ is the current averaged attention distribution and λ is the decay factor. The configuration files set $\lambda = 0.75$. Chat-template tokens are masked by default. In the main setting, candidate copied spans are restricted to the original context. The token-source ablation additionally allows candidate positions over the full replay prompt; in both settings, the replayed scaffold conditions the model state from which query-token attention is read out.

Token selection and sentence recovery. After scoring, the implementation applies top-K, top-p, or hybrid selection to obtain token positions. These selection rules follow prior token-importance methods and are not specific to RECONTEXT. The method then decodes the prompt, finds sentence boundaries, and copies the sentence containing each selected token. Empty strings and trailing special markers are removed. When multiple selected tokens fall in the same sentence, the sentence is kept once.

Recursive evidence sifting. For R replay rounds, the implementation repeats evidence proposal and sentence recovery on the current replay prompt. Newly recovered sentences are deduplicated against previously inserted sentences, accumulated into a single ordered scaffold, and inserted back between the original context and the question-side prompt. The final answer is generated only after the last replay round. The main scripts use $R = 2$ with sentence wrapping disabled, so the inserted scaffold is a compact list of copied evidence sentences.

Table 6: Ablation on recursive evidence-sifting rounds. Multiple rounds substantially improve over a single round, while the best depth varies by dataset.

R **Rounds**	**NQ 128K**		**PopQA 128K**		**InfMC 128K**
	Acc	F1	Acc	F1	Acc
1	0.04	0.21	0.01	0.10	0.48
2	0.08	0.25	**0.07**	**0.19**	0.50
3	**0.09**	0.25	0.05	0.18	0.51
4	**0.09**	**0.25**	0.05	0.17	**0.54**

Caching and length accounting. To avoid unnecessary recomputation, the implementation snapshots the key-value cache at the end of the original context. During replay generation, it restores this context cache and processes only the inserted evidence plus the question-side tokens before decoding the answer. The generation length budget is extended by the number of inserted replay tokens so that adding evidence does not reduce the maximum number of answer tokens.

C Experimental Settings

We use the same backbone model, context budget, prompting format, and answer decoding settings for RECONTEXT and the corresponding baselines within each comparison. For Qwen3 models evaluated beyond their native context window, we enable YaRN rope scaling. The main benchmark comparison and most ablations use 128K contexts; the shorter-context robustness study in Table 3 uses 64K contexts. Thinking mode is disabled unless explicitly stated, with Table 2 serving as the thinking-enabled robustness setting. All task scores in the experimental tables are reported as fractions in $[0, 1]$, while average rank in Table 1 remains on the original rank scale.

Dataset-specific maximum answer lengths and stopping behavior follow the evaluation scripts in the released code. The appendix ablations below isolate two hyperparameters on 128K NQ, PopQA, and InfBench MC: Table 6 varies the number of replay rounds R, and Table 7 varies the top-K evidence-token candidate budget while keeping the other generation settings fixed.

Table 7: Ablation on the evidence candidate budget. Larger candidate sets improve PopQA and InfMC, but can hurt NQ, revealing a recall–noise trade-off.

Top-K	NQ 128K		PopQA 128K		InfMC 128K
	Acc	F1	Acc	F1	Acc
1	0.03	0.22	0.04	0.14	0.52
8	**0.08**	**0.25**	0.07	0.19	0.50
16	**0.08**	0.25	0.05	0.17	0.55
32	0.04	0.23	**0.10**	**0.21**	**0.58**

D Additional Visualizations

The extended robustness results are reported numerically in Tables 2 and 3. These tables complement the hyperparameter visualizations in Figure 3: Table 2 evaluates whether evidence replay remains useful when thinking is enabled at 128K context length, while Table 3 checks whether the same behavior holds under a shorter 64K context budget.

E Theoretical Analysis

E.1 Theoretical Setup

Following prior works (Nichani et al., 2025; Olsson et al., 2022), we formulate the long-context task as follows. Suppose that each token $i = 1, \ldots, n$ in the context has mutually orthogonal embedding $c_i \in \mathbb{R}^d$ with $\|c_i\|_2 = 1$ and that the answer has embedding $y \in \mathbb{R}^d$. Initially, the prompt sequence is $x^{(0)} = [c_1, \ldots, c_n]$. Let $q \in \mathbb{R}^d$ denote the query embedding. The initial attention scores are:

$$a^{(0)} = \mathrm{softmax}\left([\langle x_i^{(0)}, q\rangle]_{i=1}^n\right). \tag{14}$$

The initial hidden embedding $h^{(0)}$ is:

$$h^{(0)} = \sum_{i=1}^n a_i^{(0)} x_i^{(0)}. \tag{15}$$

In each step $j \geq 1$, we first append the most relevant evidence to the sequence:

$$x^{(j)} = \left[x^{(j-1)}, x^{(j-1)}_{\mathrm{argmax}(a^{(j-1)})}\right]. \tag{16}$$

Then, we update the attention scores:

$$a^{(j)} = \mathrm{softmax}\left([\langle x_i^{(j)}, h^{(j-1)}\rangle]_{i=1}^{n+j}\right). \tag{17}$$

Finally, we update the hidden embedding:

$$h^{(j)} = \sum_{i=1}^{n+j} a_i^{(j)} x_i^{(j)}. \tag{18}$$

We assume that the context is relevant to the answer y: there exists i^* such that $y = c_{i^*}$. We also assume that the query q is relevant to the answer y: $\langle y, q\rangle > \langle c_i, q\rangle$ for all $i \neq i^*$ and $\max_{i\neq i^*} \frac{\langle y - c_i, q\rangle}{a_{i^*}^{(0)} - a_i^{(0)}} < 1$.

E.2 Proof of Theorem 1

Proof. To prove that the cosine similarity $\cos(h^{(j)}, y)$ strictly increases with each step $j \geq 1$, we will trace the evolution of the attention weights for each token.

W.l.o.g., suppose that $i^* = 1$, so $y = c_1$. The initial sequence is $x^{(0)} = [c_1, \ldots, c_n]$. Let $w_i^{(j)}$ denote the sum of the attention weights of all copies of the token c_i in the sequence $x^{(j)}$ when computing $h^{(j)}$. Thus, we can express the hidden embedding at any step j as:

$$h^{(j)} = \sum_{i=1}^n w_i^{(j)} c_i. \tag{19}$$

Since the embeddings c_i are mutually orthogonal and have unit norm ($\|c_i\|_2 = 1$), the cosine similarity between $h^{(j)}$ and $y = c_1$ is:

$$\cos(h^{(j)}, y) = \frac{\langle h^{(j)}, c_1\rangle}{\|h^{(j)}\|_2 \|c_1\|_2} \tag{20}$$

$$= \frac{w_1^{(j)}}{\sqrt{\sum_{m=1}^n (w_m^{(j)})^2}} \tag{21}$$

$$= \left(1 + \sum_{i\neq 1} \left(\frac{w_i^{(j)}}{w_1^{(j)}}\right)^2\right)^{-1/2}. \tag{22}$$

Let $R_i^{(j)} = \frac{w_1^{(j)}}{w_i^{(j)}}$. To prove that $\cos(h^{(j)}, y) > \cos(h^{(j-1)}, y)$, it is sufficient to prove that the ratio $R_i^{(j)} > R_i^{(j-1)}$ for all $i \neq 1$ and for all $j \geq 1$. We will use induction to prove this.

At $j = 0$, the attention weights are $w_i^{(0)} = a_i^{(0)}$. The initial ratio for the base state (as dictated by the dot product) can be defined as $R_i^{(0)} = e^{\langle c_1 - c_i, q\rangle}$. Since $\langle y - c_i, q\rangle < a_1^{(0)} - a_i^{(0)}$, then

$$\ln R_i^{(0)} < w_1^{(0)} - w_i^{(0)}. \tag{23}$$

We are also given that $\langle y, q\rangle > \langle c_i, q\rangle$ for all $i \neq 1$, meaning $a_1^{(0)} > a_i^{(0)}$. Thus, the most relevant evidence token at step 0 is c_1, and $x^{(1)}$ appends c_1. At step 1, the sequence has 2 copies of c_1 and 1 copy of c_i. The unnormalized attention score for

each copy of c_k is $e^{\langle c_k, h^{(0)}\rangle} = e^{w_k^{(0)}}$. Normalizing these gives:

$$w_1^{(1)} = \frac{2e^{w_1^{(0)}}}{Z_1}, \quad w_i^{(1)} = \frac{e^{w_i^{(0)}}}{Z_1}, \tag{24}$$

where $Z_j = \sum_{i=1}^n w_i^{(j)}$ is the denominator of softmax in attention scores. Evaluating the ratio $R_i^{(1)}$:

$$R_i^{(1)} = \frac{w_1^{(1)}}{w_i^{(1)}} = 2e^{w_1^{(0)} - w_i^{(0)}}. \tag{25}$$

We want to show $R_i^{(1)} > R_i^{(0)}$. Substituting our assumption $\ln R_i^{(0)} < w_1^{(0)} - w_i^{(0)}$:

$$R_i^{(1)} = 2e^{w_1^{(0)} - w_i^{(0)}} > 2e^{\ln R_i^{(0)}} = 2R_i^{(0)} > R_i^{(0)}. \tag{26}$$

Thus, $R_i^{(1)} > R_i^{(0)}$, which implies $\cos(h^{(1)}, y) > \cos(h^{(0)}, y)$.

Assume for step $j-1$ that $R_i^{(j-1)} > R_i^{(j-2)} > \cdots > R_i^{(0)} > 1$. Because $R_i^{(j-1)} > 1$, we have $w_1^{(j-1)} > w_i^{(j-1)}$. Therefore, individual tokens c_1 continue to command the highest attention scores, meaning c_1 is consistently appended. At step j, there are $N_1^{(j)} = j+1$ copies of c_1 and 1 copy of each c_i.

Let $\Delta_i^{(j)} = w_1^{(j)} - w_i^{(j)}$. The weights update according to:

$$w_1^{(j)} = \frac{(j+1)e^{w_1^{(j-1)}}}{Z_j}, \quad w_i^{(j)} = \frac{e^{w_i^{(j-1)}}}{Z_j}. \tag{27}$$

Dividing the two yields the recurrence relation for R:

$$R_i^{(j)} = (j+1)e^{\Delta_i^{(j-1)}}. \tag{28}$$

We want to prove $R_i^{(j)} > R_i^{(j-1)}$. Substituting the recurrence for $R_i^{(j-1)} = je^{\Delta_i^{(j-2)}}$, the condition becomes:

$$(j+1)e^{\Delta_i^{(j-1)}} > je^{\Delta_i^{(j-2)}}, \tag{29}$$

which is equivalent to:

$$\Delta_i^{(j-1)} > \Delta_i^{(j-2)} - \ln\left(1 + \frac{1}{j}\right). \tag{30}$$

We will in fact prove a strictly stronger statement: $\Delta_i^{(j)} > \Delta_i^{(j-1)}$ for all $j \geq 1$.

By writing $w_1^{(j)}$ and $w_i^{(j)}$ explicitly using the sum Z_j, we can express $\Delta_i^{(j)}$ as:

$$\Delta_i^{(j)} = \frac{j+1-e^{-\Delta_i^{(j-1)}}}{j+1+\Sigma^{(j-1)}}, \tag{31}$$

where $\Sigma^{(j-1)} = \sum_{m\neq 1} e^{-\Delta_m^{(j-1)}}$. Let $u = \Delta_i^{(j-1)}$ and $v = \Delta_i^{(j-2)}$. We evaluate the difference $u - \Delta_i^{(j)}$:

$$u - \Delta_i^{(j)} = u - \frac{j+1-e^{-u}}{j+1+\Sigma^{(j-1)}} \tag{32}$$

$$= \frac{u(j+1+\Sigma^{(j-1)}) - (j+1) + e^{-u}}{j+1+\Sigma^{(j-1)}}. \tag{33}$$

Let's analyze the numerator $u(j+1+\Sigma^{(j-1)}) - (j+1) + e^{-u}$. From our previous step, u is formulated as $u = \frac{j-e^{-v}}{j+\Sigma^{(j-2)}}$. By the inductive hypothesis $\Delta_m^{(j-1)} > \Delta_m^{(j-2)} - \ln(1+\frac{1}{j})$, we have $e^{-\Delta_m^{(j-1)}} < \frac{j+1}{j}e^{-\Delta_m^{(j-2)}}$. Summing this over all $m \neq 1$ gives bounds on Σ:

$$\Sigma^{(j-1)} < \frac{j+1}{j}\Sigma^{(j-2)}. \tag{34}$$

Using this upper bound, we can bound the term $u(j+1+\Sigma^{(j-1)})$ in the numerator:

$$u(j+1+\Sigma^{(j-1)}) \tag{35}$$

$$< \left(\frac{j-e^{-v}}{j+\Sigma^{(j-2)}}\right)\left(j+1+\frac{j+1}{j}\Sigma^{(j-2)}\right) \tag{36}$$

$$= \left(\frac{j-e^{-v}}{j+\Sigma^{(j-2)}}\right)\frac{j+1}{j}\left(j+\Sigma^{(j-2)}\right) \tag{37}$$

$$= \frac{j+1}{j}(j-e^{-v}) = j+1-\frac{j+1}{j}e^{-v}. \tag{38}$$

Substituting this back into the numerator of our difference expression, we have that the numerator is smaller than:

$$< \left(j+1-\frac{j+1}{j}e^{-v}\right) - (j+1) + e^{-u} \tag{39}$$

$$= e^{-u} - \frac{j+1}{j}e^{-v}. \tag{40}$$

Because we assumed $\Delta_i^{(j-1)} > \Delta_i^{(j-2)} - \ln(1+\frac{1}{j})$, it is an algebraic consequence that $e^{-u} < \frac{j+1}{j}e^{-v}$. Therefore, the numerator is < 0, which implies $u - \Delta_i^{(j)} < 0$. It follows that

$$\Delta_i^{(j)} > \Delta_i^{(j-1)}. \tag{41}$$

Because $\Delta_i^{(j)}$ is strictly increasing, it easily satisfies the required bound $\Delta_i^{(j)} > \Delta_i^{(j-1)} - \ln(1 + \frac{1}{j+1})$. This guarantees that:

$$R_i^{(j+1)} > R_i^{(j)}. \tag{42}$$

Since the ratio of the correct answer's weight to every incorrect answer's weight strictly increases at every step j, the relative mass of $w_1^{(j)}$ continuously approaches 1. Consequently, the denominator in the cosine similarity formula strictly shrinks, yielding:

$$\cos(h^{(j)}, y) > \cos(h^{(j-1)}, y), \quad \forall j \geq 1. \quad \square$$

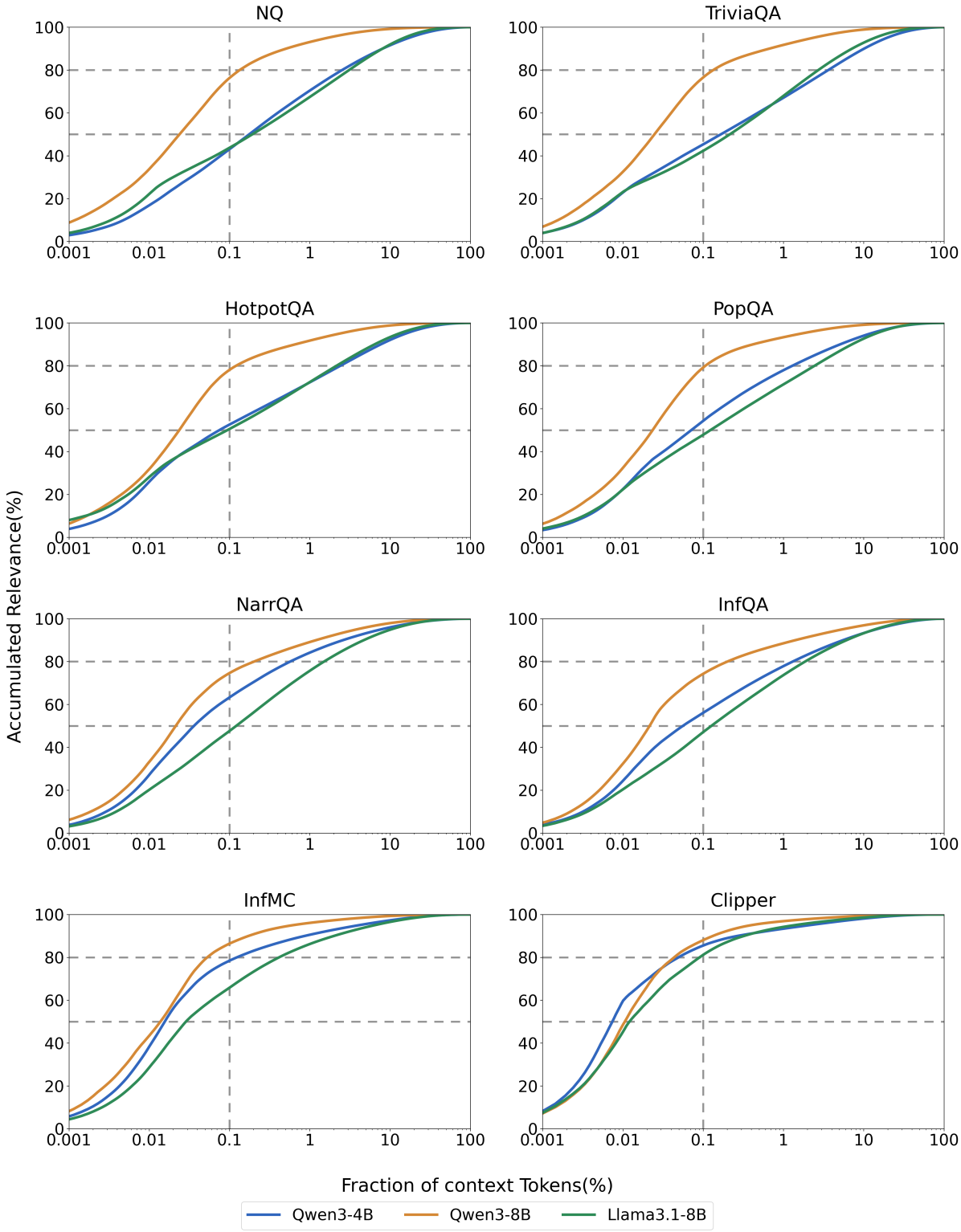

Figure 6: **Top 0.1% of context tokens already accounts for about 50% / 80% accumulated relevance score across three LLMs, corresponding to only 128 tokens in a 128K-token context.** This figure ranks all context tokens by their relevance scores with respect to the question and shows how much accumulated relevance score is covered by the top-ranked tokens. Each curve represents the mean trend over eight datasets, and the shaded region shows the variance across datasets.