Chain-of-Extractions: Iterative Evidence Memory from Attention for Long-Context QA

Anonymous ACL submission

Abstract

Long-context language models can ingest millions of tokens, yet they still struggle to assemble the correct evidence when relevant facts are scattered among distractors or depend on previously retrieved information. Chain-of-thought prompting addresses this by generating intermediate reasoning, but it is expensive, model-dependent, and its traces are not always faithful. Decode-time steering methods avoid generation but alter the model's internal dynamics, that could lead to unintended behavior.

We propose Chain-of-Extractions (CoE), an extractive method that constructs a compact, question-specific evidence memory directly from the context. CoE encodes the long input once, selects a small set of evidence anchors using query-focused retrieval heads, and then performs a covering anchor-conditioned expansion to extract dependent facts while reusing the cached key-value states and without modifying decoding. Because every item in the resulting memory is copied verbatim from the input, the working memory is fully grounded and auditable. On structured multi-hop tasks, CoE provides more consistent performance than chain-of-thought as context length grows and on four long-QA benchmarks, CoE is competitive on all context lengths while being significantly faster than reasoning methods.

1 Introduction

Current LLMs can process millions of tokens, yet simply accepting a long input does not guarantee the model will use the right parts for a given query. Early work showed strong position sensitivity with significant performance drop in the middle of the context (Liu et al., 2024). Although modern models largely overcome that bias (Pham et al., 2026), they still fail to prioritise relevant evidence when distractors are present. Recent evaluations show that models achieve near-perfect accuracy on simple needle-in-a-haystack (NIAH) tasks but struggle with multi-hop tracing and aggregation (Hsieh et al., 2024), and synthetic NIAH tests do not correctly reflect actual downstream performance (Yen et al., 2025). Hence, the main challenge is not just whether a model can access a distant token. Instead, the real issue is whether it can construct and use the correct evidence set when the relevant information is distributed, indirect, or hidden among distractors.

This challenge cannot be solved with a single perfect retrieval step, which then converts the task to a short-context problem. In many long-context tasks, relevance of a fact depends on what has already been found. Recent dynamic retrieval methods also point this out (Ye et al., 2025; Zhu et al., 2025). Similar patterns appear in other domains (Wu et al., 2025a; Li et al., 2026; Lu et al., 2026b; Opsahl-Ong et al., 2026). Hence, effective long-context use requires building a question-specific evidence state: a compact, dynamically updated set of evidence whose relevance depends on previously selected information.

Building such a state is difficult for several reasons. Noisy contexts degrade even reasoning-specialised models (Lee et al., 2026). As context length increases, the signal from relevant evidence gets diluted by many irrelevant tokens (Bansal et al., 2026). Models sometimes internally locate useful information but fail to use it correctly for the final answer (Sainath, 2026) and performance drops when the match between query and context is semantic rather than lexical (Modarressi et al., 2025).

Related Work

Inference-time reasoning is a common remedy. Techniques like chain-of-thought prompting decompose the question, track intermediate steps, and revise partial hypotheses. Yu et al. (2025b) note that adding sufficient reasoning steps can resolve many long-context retrieval failures. However, gen-

erating these steps over long contexts is computationally expensive. Furthermore, Chen et al. (2026) observe that this approach mainly benefits natively trained reasoning models. Standard instruction-tuned models do not show consistent gains from reasoning prompts. Generated reasoning traces are not always faithful. They can omit important cues (Chen et al., 2025b), rationalize post hoc (Arcuschin et al., 2025), or accumulate hallucinated intermediate states (Lu et al., 2026a).

Internal attention-based retrieval signals have been harnessed to guide evidence selection in another direction of work. For instance, Wu et al. (2025b) identify a sparse subset of attention heads that help in finding relevant information from distant context. Further, Zhang et al. (2025a) aggregate query-focused attention mass as a retrieval score. Qiu et al. (2025) show that the model must filter and de-noise the context before reasoning over a large prompted knowledge base. Attrieval (Zhang et al., 2025b) highlights that attention from reasoning tokens can locate relevant facts not mentioned in the generated text. Together, these works show that attention and KV-cache states can reliably indicate which parts of the context matter for the current question.

Decode-time steering is a common strategy to harness these internal retrieval signals. CAFE (Peng et al., 2025) performs coarse-to-fine filtering and then steer attention toward relevant content. Dysco (Ye et al., 2026) uses retrieval heads during decoding to identify and explicitly up-weight task-relevant tokens. Spotlight (Venkateswaran and Contractor, 2026) similarly modifies attention to focus more on selected prompt tokens. However, all these methods change the internal decoding mechanism of the model. Also, the above approaches are not designed for multi-hop evidence sets. Some recent remedies are: Sun et al. (2025) considers each path as an evidence set. To avoid early mistakes, Zhang et al. (2024) keep multiple partial hypotheses. Further, dynamic retrieval methods explicitly deal with interdependent subquestions (Ye et al., 2025; Zhu et al., 2025).

Our Approach

We propose Chain-of-Extractions (CoE), a method that constructs an explicit, extractive evidence memory from the context using attention signals and without modifying decoding or generating intermediate text. We encode the long context once and cache its key-value states. Extraction probes obtained from the user query then reuse this cache to score candidate evidence units. CoE first selects a small set of evidence anchors, then performs an anchor-conditioned expansion to discover a covering set of evidence. Because every item in the resulting memory is copied verbatim from the input, the intermediate state is fully grounded and auditable. Hence, our main goal is not to replace reasoning, but to make the construction of a compact, question-specific evidence state explicit and measurable.

We evaluate CoE on structured multi-hop tasks (Graphwalks, Path Walking) and on RAG QA benchmarks from HELMET (Yen et al., 2025). On the structured tasks, CoE degrades more gracefully than CoT as context length grows. On RAG QA, CoE ranks first or second across twelve model-averaged splits while adding modest latency over the vanilla baseline.

2 Extractive Evidence Memory

Given a long context x and a query q, our goal is to build an explicit evidence memory before generating the answer. We view the context as a set of evidence units $\mathcal{E}(x) = \{e_1, e_2, \ldots, e_n\}$, where each e_i is a task-specific token span (a sentence, a passage, or a structured record such as a graph edge). From these units, CoE constructs an ordered evidence memory $\mathcal{M}(q, x) = [m_1, m_2, \ldots, m_r], \quad m_i \in \mathcal{E}(x)$, which is then provided alongside q for answer generation. Because every item in $\mathcal{M}$ is copied verbatim from the input, the intermediate state is fully grounded and auditable.

We assume the long context has already been encoded and its key-value (KV) cache is available. The method proceeds in three stages: (i) score evidence units against the task query and select a small anchor set, (ii) expand the evidence by scoring units conditioned on the anchors, and (iii) assemble the final memory and generate the answer. We describe each stage below. Although our current evaluation uses a single expansion pass (two passes total), the anchor-then-expand loop can be repeated: anchors from pass I become conditioning context for pass $I+1$, stopping when no new high-scoring evidence is found or an evidence budget is reached.

Figure 1: **Multi-pass extractive evidence construction in CoE.** In the first hop, query-conditioned attention identifies high-scoring anchor evidences connected to the source or destination entities. In the second hop, these anchors are appended to the query, forming an anchor-conditioned query that enables retrieval of intermediate evidences. Using the attention graph induced by this updated query, additional evidences are extracted with respect to the first-hop anchors.

2.1 Scoring Evidence Units

We perform evidence selection using extraction probes that attend to the cached context. Initially, the probe is simply the main task query q. Let p represent an extraction probe. We use a fixed set of query-focused retrieval heads (Zhang et al., 2025a), denoted by $\mathcal{H}_{\text{QR}}$. For any given head $h \in \mathcal{H}_{\text{QR}}$, let $A_p^h(u,t)$ be the attention weight from a token position u in the extraction query to a token position t in the cached context. We average across heads:

$$A_p(u,t) = \frac{1}{|\mathcal{H}_{\text{QR}}|} \sum_{h \in \mathcal{H}_{\text{QR}}} A_p^h(u,t).$$

Next, given a set of probe token positions T, we score evidence unit e_i by averaging the attention it gets from the probe:

$$\alpha_T(e_i) = \frac{1}{|\text{span}(e_i)|} \sum_{t \in \text{span}(e_i)} \frac{1}{|T|} \sum_{u \in T} A_p(u,t).$$

To calibrate these scores, we follow earlier works on attention-score calibration (Chen et al., 2025a; Zhang et al., 2025a) and compute a residual score. Let T_{base} be the token positions of a fixed prefix in the scoring probe. The residual score is:

$$\bar{\alpha}_T(e_i) = \alpha_T(e_i) - \alpha_{T_{\text{base}}}(e_i). \quad (1)$$

All subsequent selection decisions are based on $\bar{\alpha}$.

2.2 Anchor Selection

In the first pass, we select a small set of evidence anchors based on the original query and compute $\bar{\alpha}_T(e_i)$ for every evidence unit with T as q. Let μ and σ denote the mean and standard deviation of these scores over the candidate pool. We adaptively select all units with $\bar{\alpha}_T(e_i) > \mu + \lambda\sigma$ as the anchor set $\mathcal{A} = \{a_1, \ldots, a_k\}$. These anchors serve as starting points for the subsequent expansion step: in multi-hop settings, an anchor typically identifies an initial endpoint, entity, or local fact directly matching the query. Further evidence becomes retrievable once this anchor is part of the scoring probe.

2.3 Shared Anchor-Conditioned Expansion

We concatenate the task query and all anchors into a single scoring probe: $p_{\mathcal{A}} = [q; a_1; a_2; \ldots; a_k]$. A single forward pass over the cached context yields attention from every token in $p_{\mathcal{A}}$ to every context position. Initially, we had implemented a second pass of selection by aggregating the attention scores from all of $p_{\mathcal{A}}$, and did not obtain useful retrievals. This led us to our current coverage focused retrieval where we separately retrieve matches for each $a_j \in \mathcal{A}$.

We score each evidence unit with respect to each anchor a_j by restricting the probe-token positions to those occupied by a_j. Let $T(a_j)$ denote these positions. The anchor-conditioned score of evidence unit e_i as: $s_j(e_i) = \bar{\alpha}_{T(a_j)}(e_i)$. We obtain a separate ranking of evidence units for each anchor a_j as $R_j = \mathrm{rank}_{e_i \in \mathcal{E}(x)}\, s_j(e_i)$. Different anchors can expose different parts of the context, so per-anchor ranking recovers evidence that the task query alone would miss.

2.4 Assembling the Evidence Memory

For each anchor a_j, we apply the same adaptive rule used in anchor selection: retain candidates whose score $s_j(e_i)$ exceeds $\mu_j + \lambda\,\sigma_j$, where μ_j and σ_j are computed over R_j and λ is the same coefficient as before. To guard against overly aggressive filtering, we enforce a minimum candidate count per anchor and a maximum total budget across all anchors.

Let $\mathcal{C}_j$ denote the set of retained candidates for anchor a_j. The combined evidence pool is $\mathcal{P} = \mathcal{A} \cup \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_k$. Finally, we remove any duplicate evidence units by comparing their normalized text. The final evidence memory $\mathcal{M}(q, x)$ places the anchors first, followed by ranked candidates merged across anchors in alternating order, up to the overall budget.

Algorithm 1 CoE: Evidence Extraction

Require: context x, task query q, evidence units $\mathcal{E}(x)$, threshold coefficient λ, per-anchor bounds $b_{\min}, b_{\max}$, overall budget M
Ensure: evidence memory $\mathcal{M}(q, x)$
1: Encode x once and cache its key-value states.
2: With q as scoring probe compute residual scores $\bar{\alpha}_T(e_i)$ for all $e_i \in \mathcal{E}(x)$ using retrieval heads $\mathcal{H}_{\mathrm{QR}}$.
3: Select anchors $\mathcal{A} = \{a_1, \ldots, a_k\}$: all units with $\bar{\alpha}_T(e_i) > \mu + \lambda\sigma$, subject to $b_{\min} \le k \le b_{\max}$.
4: Build the combined scoring probe $p_{\mathcal{A}} = [q;\, a_1;\, \ldots;\, a_k]$ and forward pass against the cached context.
5: **for** each anchor $a_j \in \mathcal{A}$ **do**
6: Let $T(a_j)$ be the token positions of a_j in $p_{\mathcal{A}}$.
7: Compute anchor-conditioned scores $s_j(e_i) = \bar{\alpha}_{T(a_j)}(e_i)$ from the cached attention of step 4.
8: Retain $\mathcal{C}_j = \{e_i : s_j(e_i) > \mu_j + \lambda\sigma_j\}$, enforcing at least $b_{\min}$ candidates.
9: **end for**
10: Form $\mathcal{P} = \mathcal{A} \cup \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_k$ and deduplicate.
11: Build $\mathcal{M}(q, x)$: anchors first (by descending $\bar{\alpha}$), then candidates grouped by source anchor and sorted by descending s_j, truncated to overall budget M.
12: Generate the answer using q, $\mathcal{M}(q, x)$, and cached context.

2.5 Answer Generation

Finally, we generate the actual answer using the original query and the constructed evidence memory. In this setting, the model also retains access to the original long context via the cached key-value states. Therefore, the evidence memory does not replace the context. Instead, it serves as a compact, query-specific state that highlights which context units the model should utilize. Because every memory item is an exact copy from the input, CoE introduces no generated sub-questions, reasoning chains, or synthesized facts prior to answer generation. The overall method is present in Algorithm 1.

3 Experiments

We evaluate CoE in two settings. The goal is to test whether an explicit extractive memory of evidence is a useful intermediate step for long-context reasoning.

First, we use structured reasoning tasks, Graphwalks and Path Walking, where the model has to follow a chain of records to reach the answer. These tasks let us directly check whether each method can build the needed multi-hop evidence. Second, we evaluate on the RAG subset of HELMET (Yen et al., 2025), where short-context QA examples are placed inside much longer contexts with many dis-

tractors. This setting tests whether the method can still use the relevant facts when most of the context is irrelevant.

3.1 Datasets

Structured reasoning. We use two synthetic benchmarks that explicitly require multi-step evidence aggregation: (i) *Graphwalks* (BFS): given a graph encoded as edge records, answer a breadth-first-search query over a designated start node; (ii) *Path Walking*: given a sequence of edges and requires finding a path from a start to a target node. For both tasks, each record (one edge, or one transition) is a natural unit of evidence. We evaluate at 16k and 32k context lengths. Graphwalks is scored by exact match, precision, recall, and F1; Path Walking is scored by exact-match accuracy and partial accuracy (fraction of intermediate states correctly recovered).

RAG QA. We use the four RAG tasks from HELMET: HotpotQA (Yang et al., 2018) (multi-hop reasoning), Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023). Following Yen et al. (2025), each question is presented alongside its gold context together with additional retrieved passages so that the total input reaches the target length. We evaluate at 8k, 16k, and 32k tokens, with 200 examples per split. F1 against reference answers is the evaluation metric.

3.2 Models

We use three open-weight instruction-tuned language models from two families: **Qwen3-8B**, **Qwen3-4B** (Yang et al., 2025), and **Llama-3.1-8B-Instruct** (Grattafiori et al., 2024). We additionally include **AgenticLU-8B** (Zhuang et al., 2025) as a fourth comparison point, since it is a single fine-tuned model rather than a prompting strategy, and is reported separately throughout.

3.3 Baselines

We compare CoE with five long-context QA baselines, covering the main ways inference-time or model-level reasoning is usually added.

Vanilla. The model receives the full context and the question, and directly generates the answer. This measures how much can be obtained from the base long-context model without any extra reasoning procedure.

Chain-of-Thought (CoT). The model receives the same input, but is asked to reason step by step before answering (Wei et al., 2022). This is a simple way to test whether free-form generated reasoning is enough for these tasks.

AgenticLU. We evaluate the released 8B checkpoint of AgenticLU (Zhuang et al., 2025), a model fine-tuned for long-context agentic reasoning. This baseline tests whether training a specialized long-context reasoning model closes the gap.

Tree-of-Agents (ToA). We also compare with Tree-of-Agents (Yu et al., 2025a), a multi-agent method that decomposes the question across multiple reasoning agents.

DySCO. DySCO (Ye et al., 2026) is a decoding-time method that uses retrieval heads to increase the weight of task-relevant tokens during generation. It is the closest baseline to our approach, since it also uses retrieval-head behavior, but it changes decoding directly rather than writing out an explicit evidence memory.

3.4 Implementation Details

For each example, CoE encodes the long context once and reuses the same KV cache for all extraction queries. We use the query-focused retrieval heads from Zhang et al. (2025a) ($\mathcal{H}_{\mathrm{QR}}$) directly, without doing any additional head selection. We keep the selectivity parameter fixed at $\lambda = 0.5$ in all experiments. We choose the number of anchors adaptively, capped at $k = 5$ for structured reasoning tasks and $k = 3$ for RAG QA tasks. And set the branch range to $b_{\min} = 3$ and $b_{\max} = 20$ for structured reasoning tasks, and to $b_{\min} = 3$ and $b_{\max} = 5$ for RAG QA tasks.

The evidence unit e_i is defined slightly differently across tasks. For Graphwalks and Path Walking, each line in the input, corresponding to one edge or transition, is treated as one unit. For RAG QA, we first split the document into chunks of 10 tokens and then expand each chunk to the nearest sentence boundaries. This yields passage-level evidence units while keeping the initial partitioning fine-grained. The token positions T_{base} used for residual calibration are taken from a fixed prefix of the extraction query, so they do not depend on the anchor.

Unless stated otherwise, we evaluate on $n = 200$ examples for each model and split. We report latency as the mean wall-clock time per query on

a single NVIDIA Blackwell B200 MIG instance with 90 GB VRAM. This includes the one-time context encoding cost, amortized over queries that use the same document.

Table 1: Structured reasoning results with Qwen3-8B at 16k and 32k context lengths (n=58/50 for BFS, n=200 for Path Walking). Observe that CoE provides more robust performance with increasing context lengths on both reasoning tasks.

Task BFS	Method	Exact Match	Precision	Recall	F1
16k	CoT	27.59	35.06	35.32	35.11
	DYSCO	31.58	37.44	37.75	36.64
	CoE	39.66	47.07	49.62	46.53
32k	CoT	22.00	25.77	23.72	23.77
	DYSCO	20.41	21.94	21.56	21.64
	CoE	24.00	34.33	36.71	32.95

Task Path walking	Method	Accuracy	Partial Accuracy
16k	CoT	63.00	70.38
	DYSCO	35.00	51.50
	CoE	50.50	69.25
32k	CoT	8.00	31.25
	DYSCO	1.50	18.62
	CoE	43.00	65.38

3.5 Results on Structured Reasoning

Table 1 reports results on Graphwalks and Path Walking with Qwen3-8B. CoE improves over both baselines on Graphwalks at both context lengths, with comparable gains across precision and recall.

Path Walking shows a more interesting contrast. At 16k, CoT outperforms CoE on accuracy: short transition chains fit within the model's generative working memory, and the task's step-by-step structure is well-suited to CoT. At 32k, this advantage disappears entirely. CoT's accuracy drops from 63.0 to 8.0, while CoE degrades from 50.5 to 43.0. The drop reflects a difference in how the two methods scale: CoT must regenerate and track the full transition chain at inference time, which becomes unreliable once the chain is long enough, whereas CoE materializes the relevant transitions once during extraction and then answers from that fixed memory. Where the chain is too long for in-context generation to track, an extractive evidence memory continues to work.

3.6 Results on RAG QA

On the four RAG QA tasks (Tables 2 and 5), AgenticLU underperforms the Vanilla baseline by 3–13 F1 on every model-averaged split, and DySCO by 8–22 F1. Although both methods were designed to improve long-context QA, neither matches basic prompting here.

CoT improves over Vanilla on average but is highly model-dependent. With Qwen3-8B and Qwen3-4B, it gives modest gains on HotpotQA and on a few factoid splits. With Llama-3.1-8B-Instruct, CoT degrades sharply, dropping 17.6 F1 below Vanilla on NQ-8k, 22.5 on NQ-16k, and 22.3 on NQ-32k, with comparable drops on PopQA. This matches the observation of Chen et al. (2026) that CoT primarily benefits natively-trained reasoning models.

CoE ranks first or second on all twelve model-averaged splits, with the best average rank of 1.58 (Vanilla: 1.92, CoT: 2.83, AgenticLU: 3.83, DySCO: 4.83). It wins outright on five splits, ties or trails Vanilla narrowly on four (most at 32k), and trails CoT on the three HotpotQA splits by 1–3 F1. HotpotQA is the only task in the suite with explicit two-hop structure, which favors CoT's step-by-step decomposition.

3.7 Analysis

To understand where CoE's errors come from on the structured-reasoning tasks, we decompose its predictions along two axes: whether the extracted memory contains all answer-supporting facts (coverage), and at which extraction pass each fact was first retrieved.

Coverage is the binding constraint on accuracy. For each example we record whether the extracted memory covers all gold answer-supporting facts, and cross-tabulate this against final-answer correctness (Table 3). When coverage is complete, CoE is correct on 66–85% of examples; when coverage is incomplete, accuracy falls to 0–17%. The implication is that error reduction in CoE is almost entirely a retrieval problem rather than a generation problem: given the right evidence, the underlying model answers reliably; without it, the model rarely recovers.

The two task families differ in how often retrieval succeeds. On Graphwalks, full-coverage rates are 44.8% at 16k and 30.0% at 32k, with per-sample answer-node recall of 21.2% and 9.6% respectively. On Path Walking the coverage picture

Table 2: QA results averaged across three models (Qwen3-8B, Llama-3.1-8B-Instruct, Qwen3-4B). All scores are F1 (n=200 per model). **Bold** = best, underline = second best. CoE is best or second best in accuracy compared to all baselines while being significantly faster than CoT. ⋆ Averaged over 2 models

	Accuracy (F1)						Latency (s/query)		
Split	Vanilla	CoT	DySCO	AgenticLU	ToA	CoE	Vanilla	CoT	CoE
hotpotqa_8k	63.87	**67.06**	51.61	50.63	58.90	65.21	0.33	4.31	0.98
hotpotqa_16k⋆	61.34	**63.13**	48.93	52.69	58.60	62.19	0.76	5.11	2.57
hotpotqa_32k	56.93	**60.65**	42.19	52.24	57.30	57.57	2.64	8.43	7.28
nq_8k	59.86	55.10	42.44	51.34	56.80	**61.93**	0.34	6.50	0.96
nq_16k	57.75	51.34	39.32	54.40	**60.80**	59.04	0.75	7.43	2.66
nq_32k	56.58	49.39	34.88	53.50	**58.10**	54.47	2.09	10.54	8.39
triviaqa_8k	87.54	82.93	79.15	79.08	86.60	**88.27**	0.33	5.87	1.02
triviaqa_16k	90.03	82.63	80.24	82.75	85.00	**90.38**	0.74	5.21	2.60
triviaqa_32k⋆	**89.88**	81.21	75.05	85.42	86.20	89.20	1.91	7.18	7.43
popqa_8k	**64.43**	57.33	46.84	53.83	53.90	63.32	0.28	7.02	0.80
popqa_16k⋆	59.54	53.25	46.38	51.55	54.40	**59.79**	0.71	7.13	2.26
popqa_32k⋆	**57.83**	50.02	39.18	49.94	52.20	57.05	1.81	8.79	7.31

is more favorable, with full-recall rates of 64.0% and 69.0% and gold fact recall above 89%. Graph-walks is the harder retrieval problem because the BFS frontier expands combinatorially, so the set of answer-supporting edges is larger and more dispersed.

Table 3: Coverage vs. correctness for CoE on structured-reasoning tasks (Qwen3-8B). *Cov.* = full coverage of answer-supporting facts. We find that the correctness of the answer is very strongly correlated with coverage of relevant edges in the extracted memory.

	Full Coverage		Missing Coverage	
Split	Correct	Wrong	Correct	Wrong
BFS 16k	22	4	1	31
BFS 32k	12	3	0	35
Path 16k	89	39	12	60
Path 32k	91	47	2	60

Intermediate hops are recovered only after anchoring. On Path Walking, we track which extraction pass first surfaces each gold transition. Hop 1 (the start state) and hop 4 (the goal state) are surfaced in pass 1 by the original query for the majority of examples, since both endpoints are mentioned in the query itself. Hops 2 and 3 are recovered almost exclusively in pass 2 (Table 4): the original query alone does not retrieve any intermediate transitions, and they are surfaced only after the pass-1 anchors are added to the extraction query.

Table 4: Number of Path Walking examples (out of 200) in which each hop is first surfaced in pass 1 vs. pass 2 of CoE's extraction. Anchors from pass 1 enable retrieval of intermediate hops in pass 2. All the intermediate edges are retrieved in the second pass.

	Path 16k		Path 32k	
Hop	Pass 1	Pass 2	Pass 1	Pass 2
1 (start)	197	2	193	7
2 (intermediate)	0	183	0	193
3 (intermediate)	0	139	0	145
4 (goal)	148	45	152	47

This pattern is a direct empirical test of the anchor-conditioned design described in Section 2: facts whose relevance depends on previously retrieved evidence are not visible to attention until that earlier evidence is added to the query. A single-pass extractor would, on this task, miss the majority of intermediate transitions and would consequently fail the chain-of-evidence requirement that the final answer depends on.

4 Conclusion

We present CoE, a long-context reasoning method that replaces generated reasoning steps with ev-

idence extracted directly from the input context. Instead of asking the model to write a chain of thought, CoE uses retrieval-head attention to select answer-relevant spans. Its two-pass design starts from the original query, then uses the extracted anchors to recover facts whose relevance depends on what was already found. On structured multi-hop tasks, CoE improves over CoT and DySCO on Graphwalks, and degrades more slowly with context length on Path Walking. The analysis shows a consistent pattern: when extracted evidence covers all answer-supporting facts, the model usually answers correctly; when it fails, the issue is that some needed evidence was never extracted, not the final reasoning step. On four RAG QA benchmarks, CoE ranks first or second across all twelve model-averaged splits while adding modest overhead over direct prompting, and being significantly faster than reasoning-based methods.

5 Limitations

CoE relies on the extracted evidence being sufficient for the question. As contexts get longer and supporting facts become more dispersed, evidence coverage becomes increasingly important. We evaluate the two-pass design on chains up to four hops. Deeper multi-hop settings may require more passes or a different anchoring strategy.

The method requires access to the model's attention weights and KV cache, so it is not applicable to black-box API-only models. It also depends on the existence of reliable query-focused retrieval heads; models that lack such heads may need a head-selection step that we do not currently perform.

References

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. 2025. Chain-of-thought reasoning in the wild is not always faithful. Preprint, arXiv:2503.08679.

Rachit Bansal, Aston Zhang, Rishabh Tiwari, Lovish Madaan, Sai Surya Duvvuri, Fnu Devvrit, David Brandfonbrener, David Alvarez-Melis, Prajjwal Bhargava, Mihir Kale, and Samy Jelassi. 2026. Let's (not) just put things in context: Test-time training for long-context LLMs. In The Fourteenth International Conference on Learning Representations.

Shijie Chen, Bernal Jimenez Gutierrez, and Yu Su. 2025a. Attention in large language models yields efficient zero-shot re-rankers. In The Thirteenth International Conference on Learning Representations.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. 2025b. Reasoning models don't always say what they think. Preprint, arXiv:2505.05410.

Ziyang Chen, Xing Wu, Junlong Jia, Chaochen Gao, Qi Fu, Debing Zhang, and Songlin Hu. 2026. Longbench pro: A more realistic and comprehensive bilingual long-context evaluation benchmark. Preprint, arXiv:2601.02872.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. RULER: What's the real context size of your long-context language models? In First Conference on Language Modeling.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1601–1611.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. Transactions of the Association for Computational Linguistics, 7:453–466.

Seongyun Lee, Yongrae Jo, Minju Seo, Moontae Lee, and Minjoon Seo. 2026. Lost in the noise: How reasoning models fail with contextual distractors. Preprint, arXiv:2601.07226.

Yifei Li, Weidong Guo, Lingling Zhang, Rongman Xu, Muye Huang, Hui Liu, Lijiao Xu, Yu Xu, and Jun Liu. 2026. Locomo-plus: Beyond-factual cognitive memory evaluation framework for llm agents. Preprint, arXiv:2602.10715.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. Transactions of the Association for Computational Linguistics, 12:157–173.

Haolang Lu, Minghui Pan, Ripeng Li, Guoshun Nan, Jialin Zhuang, Zijie Zhao, Zhongxiang Sun, Kun Wang, and Yang Liu. 2026a. Streaming hallucination detection in long chain-of-thought reasoning. Preprint, arXiv:2601.02170.

Zhiyuan Lu, Chenliang Li, Yingcheng Shi, Weizhou Shen, Ming Yan, and Fei Huang. 2026b. Corpusqa: A 10 million token benchmark for corpus-level analysis and reasoning. Preprint, arXiv:2601.14952.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers), pages 9802–9822.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schuetze. 2025. Nolima: Long-context evaluation beyond literal matching. In Forty-second International Conference on Machine Learning.

Krista Opsahl-Ong, Arnav Singhvi, Jasmine Collins, Ivan Zhou, Cindy Wang, Ashutosh Baheti, Owen Oertell, Jacob Portes, Sam Havens, Erich Elsen, Michael Bendersky, Matei Zaharia, and Xing Chen. 2026. Officeqa pro: An enterprise benchmark for end-to-end grounded reasoning. Preprint, arXiv:2603.08655.

Han Peng, Jinhao Jiang, Zican Dong, Wayne Xin Zhao, and Lei Fang. 2025. CAFE: Retrieval head-based coarse-to-fine information seeking to enhance multi-document QA capability. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 12966–12978, Suzhou, China. Association for Computational Linguistics.

Thinh Pham, Nguyen Phan Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. 2026. SealQA: Raising the bar for reasoning in search-augmented language models. In The Fourteenth International Conference on Learning Representations.

Yifu Qiu, Varun R. Embar, Yizhe Zhang, Navdeep Jaitly, Shay B. Cohen, and Benjamin Han. 2025. Eliciting in-context retrieval and reasoning for long-context large language models. In Findings of the Association for Computational Linguistics: ACL 2025, pages 3176–3192, Vienna, Austria. Association for Computational Linguistics.

Pravish Sainath. 2026. When long contexts break logic: Separating evidence use and decision bias in instruction-tuned LLMs. In ICLR 2026 Workshop on Logical Reasoning of Large Language Models.

Hao Sun, Hengyi Cai, Yuchen Li, Xuanbo Fan, Xiaochi Wei, Shuaiqiang Wang, Yan Zhang, and Dawei Yin. 2025. Enhancing retrieval-augmented generation via evidence tree search. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 24116–24127, Vienna, Austria. Association for Computational Linguistics.

Praveen Venkateswaran and Danish Contractor. 2026. Spotlight your instructions: Instruction-following with dynamic attention steering. In Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3752–3770, Rabat, Morocco. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In Advances in Neural Information Processing Systems.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. 2025a. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In The Thirteenth International Conference on Learning Representations.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2025b. Retrieval head mechanistically explains long-context factuality. In The Thirteenth International Conference on Learning Representations.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. arXiv preprint arXiv:2505.09388.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In Proceedings of the 2018 conference on empirical methods in natural language processing, pages 2369–2380.

Linhao Ye, Lang Yu, Zhikai Lei, Qin Chen, Jie Zhou, and Liang He. 2025. Optimizing question semantic space for dynamic retrieval-augmented multi-hop question answering. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 17814–17824, Vienna, Austria. Association for Computational Linguistics.

Xi Ye, Wuwei Zhang, Fangcong Yin, Howard Yen, and Danqi Chen. 2026. Dysco: Dynamic attention-scaling decoding for long-context lms. 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 models effectively and thoroughly. In The Thirteenth International Conference on Learning Representations.

Song Yu, Xiaofei Xu, Ke Deng, Li Li, and Lin Tian. 2025a. Tree of agents: Improving long-context capabilities of large language models through multi-perspective reasoning. arXiv preprint arXiv:2509.06436.

Yijiong Yu, Zhixiao Qi, Yongfeng Huang, Wei Wang, Weifeng.liu, Ran Chen, and Ji Pei. 2025b. Long-context language models fail in basic retrieval tasks without sufficient reasoning steps. In Findings of the Association for Computational Linguistics: EMNLP 2025, pages 5615–5634, Suzhou, China. Association for Computational Linguistics.

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Liu Yong, and Shen Huang. 2024. End-to-end beam retrieval for multi-hop question answering. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 1718–1731, Mexico City, Mexico. Association for Computational Linguistics.

Wuwei Zhang, Fangcong Yin, Howard Yen, Danqi Chen, and Xi Ye. 2025a. Query-focused retrieval heads improve long-context reasoning and re-ranking. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 23791–23805, Suzhou, China. Association for Computational Linguistics.

Yuwei Zhang, Jayanth Srinivasa, Gaowen Liu, and Jingbo Shang. 2025b. Attention reveals more than tokens: Training-free long-context reasoning with attention-guided retrieval. Preprint, arXiv:2503.09819.

Rongzhi Zhu, Xiangyu Liu, Zequn Sun, Yiwei Wang, and Wei Hu. 2025. Mitigating lost-in-retrieval problems in retrieval augmented multi-hop question answering. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22362–22375, Vienna, Austria. Association for Computational Linguistics.

Yufan Zhuang, Xiaodong Yu, Jialian Wu, Ximeng Sun, Ze Wang, Jiang Liu, Yusheng Su, Jingbo Shang, Zicheng Liu, and Emad Barsoum. 2025. Self-taught agentic long context understanding. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5525–5537.

A Complete Evaluation Results

Table 5: Full per-model results on RAG QA benchmarks. **Bold** = best accuracy per row. All scores are F1 (n=200). AgenticLU-8B is model-specific and reported once per split. We evaluated multiple combinations of models and datasets; however few entries are missing due to incomplete runs.

		Accuracy (F1)						Latency (s/query)		
Split	**Model**	Vanilla	CoE	CoT	DySCO	ToA	AgenticLU	Simple	CoE	CoT
	Qwen3-8B	69.07	67.11	**72.66**	56.34	60.70		0.38	0.98	4.82
hotpotqa_8k	Llama-3.1-8B	65.12	**66.13**	59.96	52.21	56.60	50.63	0.28	0.79	4.64
	Qwen3-4B	57.41	62.40	**68.55**	46.29	59.50		0.32	1.16	3.47
	Qwen3-8B	64.51	62.47	**70.61**	49.67	62.70		0.85	2.77	6.05
hotpotqa_16k	Llama-3.1-8B	61.96	**62.93**	55.97	47.65	54.40	52.69	0.65	2.28	4.41
	Qwen3-4B	57.56	61.16	**62.81**	49.47	—		0.79	2.67	4.86
	Qwen3-8B	59.11	60.30	**65.92**	44.88	63.50		3.30	7.21	11.52
hotpotqa_32k	Llama-3.1-8B	**61.88**	60.26	59.84	42.83	57.10	52.24	2.61	7.53	6.68
	Qwen3-4B	49.81	52.14	**56.19**	38.86	51.10		2.00	7.09	7.10
	Qwen3-8B	61.45	**66.11**	63.05	39.40	62.20		0.41	1.01	5.80
nq_8k	Llama-3.1-8B	62.57	**63.50**	45.02	50.34	56.00	51.34	0.29	0.92	8.95
	Qwen3-4B	55.57	56.18	**57.23**	37.58	52.20		0.33	0.96	4.74
	Qwen3-8B	61.75	**67.75**	62.62	37.43	65.30		0.87	2.92	7.23
nq_16k	Llama-3.1-8B	**61.65**	56.03	39.19	41.54	58.40	54.40	0.66	2.23	8.54
	Qwen3-4B	49.85	53.34	52.22	39.00	**58.80**		0.73	2.82	6.51
	Qwen3-8B	61.58	**65.96**	60.79	34.47	63.90		2.29	9.78	10.70
nq_32k	Llama-3.1-8B	58.14	51.00	35.80	33.61	**58.30**	53.50	1.81	5.62	11.97
	Qwen3-4B	50.01	46.46	51.58	36.56	**52.20**		2.16	9.77	8.96
	Qwen3-8B	87.57	88.02	89.00	82.64	**89.70**		0.39	1.20	5.41
triviaqa_8k	Llama-3.1-8B	88.17	**90.98**	72.51	75.61	87.60	79.08	0.29	0.81	7.97
	Qwen3-4B	86.89	85.82	**87.27**	79.19	82.60		0.30	1.05	4.23
	Qwen3-8B	**88.83**	88.45	86.48	81.77	87.90		0.86	2.80	5.86
triviaqa_16k	Llama-3.1-8B	92.00	**92.17**	77.88	78.77	85.90	82.75	0.67	2.35	4.62
	Qwen3-4B	89.25	**90.51**	83.54	80.19	81.40		0.70	2.66	5.16
	Qwen3-8B	—	—	—	—	**87.40**		—	—	—
triviaqa_32k	Llama-3.1-8B	92.50	**96.75**	82.25	77.51	85.90	85.42	1.84	7.68	6.80
	Qwen3-4B	**87.25**	81.65	80.17	72.58	85.40		1.98	7.18	7.56
	Qwen3-8B	60.17	**66.28**	64.53	53.82	57.70		0.34	0.85	6.97
popqa_8k	Llama-3.1-8B	**70.53**	62.25	48.48	46.61	52.70	53.83	0.24	0.78	8.49
	Qwen3-4B	**62.58**	61.44	58.98	40.10	51.20		0.27	0.77	5.60
	Qwen3-8B	60.17	59.83	**61.20**	49.28	56.30		0.74	2.52	6.70
popqa_16k	Llama-3.1-8B	**63.50**	59.32	45.85	48.52	—	51.55	0.57	1.90	8.62
	Qwen3-4B	54.94	**60.21**	52.70	41.34	52.40		0.82	2.35	6.08
	Qwen3-8B	**57.18**	55.08	—	44.36	52.80		2.03	8.60	—
popqa_32k	Llama-3.1-8B	**60.83**	60.73	48.41	44.58	53.80	49.94	1.61	4.83	9.24
	Qwen3-4B	**55.47**	55.35	51.63	28.61	50.00		1.80	8.49	8.34