Lost in Decomposition: Analyzing and Mitigating the Limitations of Long Context Methods via Context Dependency

Jiayuan Guo*, **Yueyang Su**†, **Yuyao Ge, Saiping Guan,**
Lei Yu, Jiafeng Guo, Xueqi Cheng
State Key Laboratory of AI Safety
Institute of Computing Technology, Chinese Academy of Sciences
University of Chinese Academy of Sciences
{guojiayuan24s, suyueyang}@ict.ac.cn
*First author. †Corresponding author.

Abstract

Long context large language models exhibit the "lost in the middle" problem, where models struggle to effectively utilize information located in the middle of long contexts. Although existing workflow-based long context methods (e.g., RAG) alleviate this problem and perform well on specific datasets, can their effectiveness generalize to all types of datasets? In this work, we systematically investigate the cross-dataset generalization of long context methods. Our evaluation reveals that these methods are not universally effective. Such substantial performance variability underscores the risks of performance degradation associated with the indiscriminate application of long context methods. We investigated the reason for the failure of long context methods. We found that the intrinsic decomposition mechanisms of long context methods hinder context dependency modeling, causing these methods to suffer performance declines on documents with strong context dependency. To address this issue, We propose **CoDaR** (**Co**ntext **D**ependency-**a**ware **R**outing), a training-free adaptive routing strategy. By analyzing the context dependency strength of documents, CoDaR adaptively invokes long context methods, thereby significantly enhancing their overall robustness across different types of datasets.

1 Introduction

Long context large language models have demonstrated remarkable capabilities across diverse natural language processing tasks. However, empirical studies reveal the "lost in the middle" problem (Liu et al., 2023), where Full-Context (feeding all input directly into long context LLMs) struggles to effectively utilize information located in the middle of long contexts compared to that at the beginning or end. To address this problem, various workflow-based methods have been proposed to enhance the long context processing capabilities of base models without altering their parameters (Liu et al., 2025). These include Retrieval-Augmented Generation (RAG) (Xu et al., 2023; Zhao et al., 2024b; Jiang et al., 2024c; Qian et al., 2025), multi-agent collaboration (Zhang et al., 2024b; Zhao et al., 2024a), memory augmentation (Zhou et al., 2023; Zhong et al., 2024; Wang et al., 2024; Chhikara et al., 2025; Xu et al., 2025), and context compression (Lee et al., 2024; Zhang et al., 2024a; Hu et al., 2025). For brevity, we refer to these workflow-based long context methods as "long context methods" in this paper.

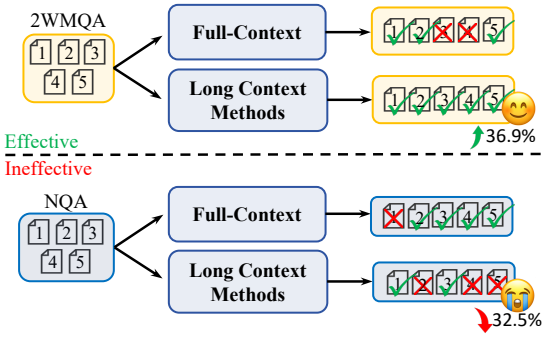


Figure 1: Long context methods are not always effective. Relative to the Full-Context baseline, a method can achieve a 36.9% performance gain on 2WikiMQA(2WMQA) while suffering a 32.5% decline on NarrativeQA(NQA).

Although existing long context methods have alleviated the "lost in the middle" problem and improved the long context processing capabilities of models on specific datasets, can their effectiveness generalize to all types of datasets? We selected representative methods encompassing RAG, multi-agent collaboration, memory augmentation, and context compression. Through evaluation across diverse datasets, we observed that long context methods are not universally effective. As illustrated in Figure 1, relative to the Full-Context baseline,

Findings of the Association for Computational Linguistics: ACL 2026, pages 42267–42285

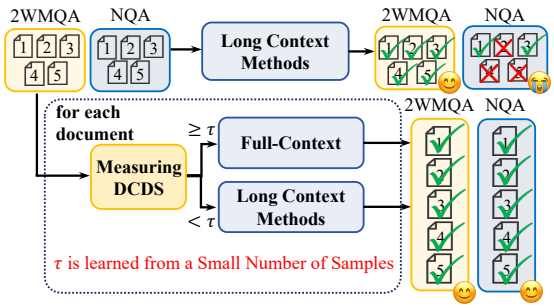


Figure 2: Overview of CoDaR framework.

the same method can achieve a 36.9% performance gain on one dataset while causing a 32.5% degradation on another. This significant performance variance of long context methods across datasets underscores the risks of performance degradation caused by indiscriminately deploying them without accounting for dataset characteristics.

We investigate the root causes of the failure of long context methods and find that the decomposition mechanisms (e.g., chunk-wise iterative processing, chunk retrieval) inherent in these methods disrupt the document's Context Dependency Structure. Although these approaches are designed to retain long-range dependency chains, they exhibit a characteristic failure mode: when documents exhibit strong context dependency, decomposition prevents the model from jointly reasoning over interdependent segments, resulting in degraded performance. In contrast, long context methods are substantially more effective for documents with weak context dependency, suggesting an intrinsic upper bound on their ability to preserve context dependency. To quantitatively investigate whether context dependency serves as the fundamental bottleneck and how it shapes the effectiveness of long-context methods, we introduce the Document Context Dependency Score (DCDS) to measure the strength of context dependency. Furthermore, through extensive experiments, we demonstrate that the effectiveness of long context methods is inversely correlated with the document's context dependency strength.

Based on these findings, to enhance the robustness of long context methods, we propose **CoDaR** (**Co**ntext **D**ependency-**a**ware **R**outing). As illustrated in Figure 2, this is a training-free adaptive routing strategy. By leveraging the Document Context Dependency Score(DCDS) as a signal, CoDaR adaptively invokes long context methods only for documents with weak context dependency, while reverting to Full-Context for those with strong dependencies. In summary, our main contributions are as follows:

- Through systematic experiments, we identify the upper bound of long context methods in handling long-range context dependency, demonstrating that such methods are not universally effective.
- We propose DCDS as a quantitative measure of document-level context dependency and empirically demonstrate an inverse relationship between context dependency strength and the effectiveness of long context methods.
- We propose **CoDaR**, a training-free adaptive routing strategy, which significantly enhances the robustness of applying long context methods.

2 Related work

Long Context Large Language Models Recent advances in Transformer architectures and computational infrastructure are enabling LLM to support increasingly long context windows, reaching lengths of up to 200K (Anthropic, 2025) or even 1M (Comanici et al., 2025). However, recent studies reveal that long context LLMs struggle to effectively use information from the middle of extended contexts, a phenomenon known as "lost in the middle" (Liu et al., 2023).

Workflow-based Long Context Methods Numerous workflow-based long context methods have been proposed to mitigate the "lost in the middle" problem. Retrieval-Augmented Generation (RAG) (Edge et al., 2024; Asai et al., 2024; Qian et al., 2025) enable models to selectively access relevant information from large context windows. Multi-agent collaboration (Zhao et al., 2024a; Zhang et al., 2024b) distribute the processing burden across multiple model instances. Context compression (Zhang et al., 2024a; Hu et al., 2025) compress less salient tokens while preserving critical information. Furthermore, memory augmentation (Zhong et al., 2024; Chhikara et al., 2025; Xu et al., 2025) employ memory modules to manage extended contextual information more effectively. Although current long context methods demonstrate superior performance on selected datasets,

existing research often overlooks their generalization capabilities. Our experiments reveal that these methods are not universally effective; we analyze their limitations and mitigate their performance degradation on unsuitable documents.

3 Preliminary

3.1 Long Context QA

The goal of long context QA can be described as follows. For a given question q and an extended context document $D = \{t_i\}_{i=1}^{n}$ where n typically exceeds 4096 tokens (Bai et al., 2023), substantially longer than conventional document-based QA, the QA system is asked to provide an answer a based on the long context D. This task can be formally described as:

$$a = f(D, q), \tag{1}$$

where f denotes the QA system being evaluated.

3.2 Full-Context

Full-Context is a long context processing approach where the entire input document D is directly fed into the long context large language model without any intermediate decomposition mechanisms. Formally, the Full-Context QA system can be expressed as:

$$a = f_{\text{full}}(D, q), \tag{2}$$

where the model processes the complete document D and question q in a single forward pass, relying solely on the model's native capability for long-context processing.

3.3 Paradigms of Long Context Methods

To comprehensively investigate long context methods, we categorize them into the following four paradigms based on their core processing mechanisms.

- **Retrieval Augmented Generation (RAG)** RAG (Zhao et al., 2024b; Jiang et al., 2024c; Asai et al., 2024; Qian et al., 2025) retrieves and processes only relevant document segments rather than entire documents, thereby enhancing the model's focus on pertinent information.

- **Multi-agent Collaboration** Multi-agent collaboration (Zhao et al., 2024a; Zhang et al., 2024b) distributes tasks across multiple agents to alleviate individual model attention burden.

- **Memory Augmentation** Memory augmentation (Wang et al., 2023; Zhong et al., 2024; Chhikara et al., 2025; Xu et al., 2025) address long context challenges by maintaining a memory module that retains important information while discarding redundancies.

- **Context Compression** Context compression (Jiang et al., 2024b; Zhang et al., 2024a; Hu et al., 2025) reduce document length by removing redundant information while preserving critical content, thereby reducing model processing burden.

4 Long Context Methods are Not Universally Effective across Datasets

To investigate whether long context methods are universally effective, we select representative methods from various paradigms and evaluate their effectiveness across different types of datasets.

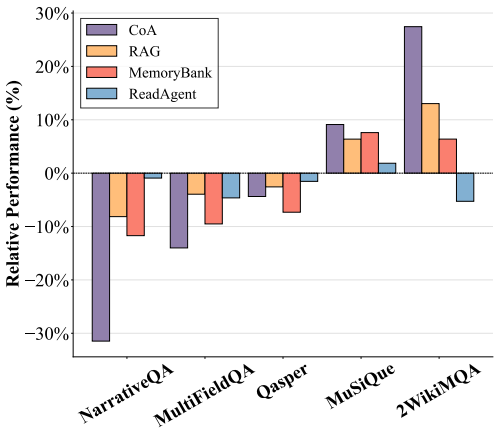


Figure 3: The average relative performance of long context methods varies across different datasets.

4.1 Experimental Setup

Datasets To ensure comprehensive coverage of diverse text types, we select five representative datasets from LongBench (Bai et al., 2023), encompassing narrative documents, academic papers, multi-domain documents, and synthetic documents. Specifically, NarrativeQA (**NQA**) (Kočiskỳ et al., 2018) contains long-form novels and movie scripts; Qasper (**QAS**) (Dasigi et al., 2021) comprises academic papers from ArXiv; MultiFieldQA (**MFQA**) (Bai et al., 2023) contains documents sourced from multiple different fields; 2WikiMultiHopQA (**2WMQA**) (Ho et al., 2020) contains

Metric	Method	NarrativeQA	MultiFieldQA	Qasper	MuSiQue	2WikiMQA
F1	Full	$33.62_{(-)}$	$54.17_{(-)}$	$49.42_{(-)}$	$39.29_{(-)}$	$55.37_{(-)}$
	RAG	$29.68_{(\downarrow 11.72)}$	$53.27_{(\downarrow 1.66)}$	$50.51_{(\uparrow 2.21)}$	$39.69_{(\uparrow 1.02)}$	$61.45_{(\uparrow 10.98)}$
	CoA	$26.20_{(\downarrow 22.06)}$	$45.06_{(\downarrow 16.82)}$	$43.09_{(\downarrow 12.81)}$	$50.10_{(\uparrow 27.51)}$	$70.96_{(\uparrow 28.16)}$
	MemoryBank	$28.68_{(\downarrow 14.69)}$	$50.56_{(\downarrow 6.66)}$	$47.28_{(\downarrow 4.33)}$	$40.68_{(\uparrow 3.54)}$	$56.39_{(\uparrow 1.84)}$
	ReadAgent	$33.31_{(\downarrow 0.92)}$	$51.66_{(\downarrow 4.63)}$	$48.66_{(\downarrow 1.54)}$	$40.02_{(\uparrow 1.86)}$	$52.45_{(\downarrow 5.27)}$
ROUGE	Full	$32.94_{(-)}$	$53.17_{(-)}$	$48.31_{(-)}$	$39.29_{(-)}$	$55.27_{(-)}$
	RAG	$29.55_{(\downarrow 10.29)}$	$51.80_{(\downarrow 2.58)}$	$49.05_{(\uparrow 1.53)}$	$39.57_{(\uparrow 0.71)}$	$61.29_{(\uparrow 10.89)}$
	CoA	$26.03_{(\downarrow 20.98)}$	$43.48_{(\downarrow 18.23)}$	$41.20_{(\downarrow 14.72)}$	$49.75_{(\uparrow 26.62)}$	$70.64_{(\uparrow 27.71)}$
	MemoryBank	$28.42_{(\downarrow 13.72)}$	$49.70_{(\downarrow 6.53)}$	$45.85_{(\downarrow 5.09)}$	$40.69_{(\uparrow 3.56)}$	$56.25_{(\uparrow 1.77)}$
	ReadAgent	$32.88_{(\downarrow 0.18)}$	$51.20_{(\downarrow 3.70)}$	$47.54_{(\downarrow 1.59)}$	$39.75_{(\uparrow 1.17)}$	$52.31_{(\downarrow 5.36)}$

Table 1: Comparison of long context methods on five datasets. We evaluate Full-Context (Full), CoA, MemoryBank, RAG, and ReadAgent using F1 and ROUGE-L metrics. Numbers in parentheses represent relative performance differences (in %) relative to Full-Context (Full), where ↑ indicates improvements and ↓ indicates degradation.

documents composed of multiple Wikipedia passages; and MuSiQue (**MSQ**) (Trivedi et al., 2022) contains documents composed of passages containing supporting evidence and irrelevant passages.

Evaluated Long Context Methods To ensure a comprehensive evaluation, we select one representative method for each of the four paradigms described in Section 3.3. Specifically, we adopt Naive RAG (Xu et al., 2023) for RAG. For Multi-agent Collaboration, we employ Chain of Agents (CoA) (Zhang et al., 2024b). Memory Augmentation is represented by MemoryBank (Zhong et al., 2024), and Context Compression is evaluated using ReadAgent (Lee et al., 2024). Detailed descriptions and experimental configurations for each method are provided in Appendix A.

Models Unless otherwise specified, we use GPT-4O-MINI-2024-07-18 (Menick et al., 2024) as our foundation model. To ensure the reliability of our findings, we further evaluate several representative models, including PHI-3-MINI-128K-INSTRUCT (Abdin et al., 2024), QWEN2.5-3B-INSTRUCT, and QWEN2.5-7B-INSTRUCT (Qwen et al., 2025). For all evaluated models, we set the temperature to 0 to ensure deterministic results.

Metrics We employ F1 score and ROUGE-L to evaluate the absolute performance of the long context methods. To assess the effectiveness of these methods, we introduce a relative performance metric, denoted as Δ_M. Specifically, we establish the Full-Context setting as the baseline, which represents the model's native capability for long context processing. The effectiveness of each method is then quantified by calculating the relative performance change relative to this baseline:

$$\Delta_M = \frac{\text{Score}_M - \text{Score}_{\text{full}}}{\text{Score}_{\text{full}}} \times 100\%, \quad (3)$$

where Score_M represents the performance of the specific long context method, and $\text{Score}_{\text{full}}$ represents the performance of Full-Context. $\Delta_M > 0$ indicates that the method enhances the model's long context processing capability (effective), while $\Delta_M < 0$ implies a reduction in capability (ineffective).

4.2 Results and Analysis

As presented in Table 1, we find that the effectiveness of long context methods is inconsistent across different datasets. For instance, CoA achieves a +27.5% F1 improvement over Full-Context on MuSiQue, yet suffers a 22.1% decline on NarrativeQA. The ROUGE-L metrics demonstrate consistent trends with F1 scores.

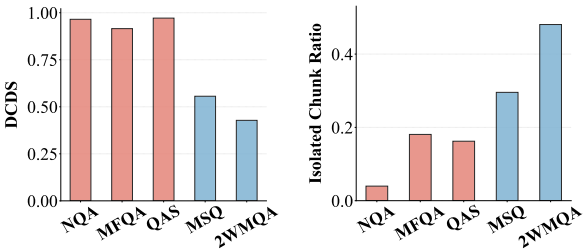
To investigate whether the effectiveness of long context methods is correlated with foundation models' intrinsic capabilities, we conducted experiments across the four foundation models. As shown in Table 2, whether long context methods are effective or ineffective is generally consistent across different models. For instance, with the CoA method, Phi-3-mini demonstrates a 36.9% improvement relative to Full-Context on 2WikiMQA but a 32.5% decline on NarrativeQA. For Qwen2.5-3B, these figures are 30.7% and 31.7%, respectively. A similar phenomenon is observed across other evaluated

Model	Method	NarrativeQA	MultiFieldQA	Qasper	MuSiQue	2WikiMQA
Phi-3-mini	Full	25.52$_{(-)}$	53.84$_{(-)}$	39.96$_{(-)}$	25.49$_{(-)}$	36.19$_{(-)}$
	RAG	23.38$_{(\downarrow 8.4)}$	53.21$_{(\downarrow 1.2)}$	39.27$_{(\downarrow 1.7)}$	26.89$_{(\uparrow 5.5)}$	40.31$_{(\uparrow 11.4)}$
	CoA	17.23$_{(\downarrow 32.5)}$	43.65$_{(\downarrow 18.9)}$	37.70$_{(\downarrow 5.7)}$	28.62$_{(\uparrow 12.3)}$	49.54$_{(\uparrow 36.9)}$
	MemoryBank	24.94$_{(\downarrow 2.3)}$	51.54$_{(\downarrow 4.3)}$	36.56$_{(\downarrow 8.5)}$	25.43$_{(\downarrow 0.2)}$	39.19$_{(\uparrow 8.3)}$
Qwen2.5-3b	Full	22.71$_{(-)}$	49.13$_{(-)}$	36.82$_{(-)}$	20.82$_{(-)}$	36.39$_{(-)}$
	RAG	22.62$_{(\downarrow 0.4)}$	44.91$_{(\downarrow 8.6)}$	34.03$_{(\downarrow 7.6)}$	23.76$_{(\uparrow 14.1)}$	42.86$_{(\uparrow 17.8)}$
	CoA	15.52$_{(\downarrow 31.7)}$	46.43$_{(\downarrow 5.5)}$	38.37$_{(\uparrow 4.2)}$	21.44$_{(\uparrow 3.0)}$	47.55$_{(\uparrow 30.7)}$
	MemoryBank	18.77$_{(\downarrow 17.3)}$	39.73$_{(\downarrow 19.1)}$	33.02$_{(\downarrow 10.3)}$	23.86$_{(\uparrow 14.6)}$	38.81$_{(\uparrow 6.6)}$
Qwen2.5-7b	Full	29.30$_{(-)}$	52.51$_{(-)}$	43.50$_{(-)}$	30.36$_{(-)}$	46.33$_{(-)}$
	RAG	25.76$_{(\downarrow 12.1)}$	50.22$_{(\downarrow 4.4)}$	42.11$_{(\downarrow 3.2)}$	31.84$_{(\uparrow 4.9)}$	51.88$_{(\uparrow 12.0)}$
	CoA	17.68$_{(\downarrow 39.7)}$	44.74$_{(\downarrow 14.8)}$	42.09$_{(\downarrow 3.2)}$	28.44$_{(\downarrow 6.3)}$	52.85$_{(\uparrow 14.1)}$
	MemoryBank	25.63$_{(\downarrow 12.5)}$	48.32$_{(\downarrow 8.0)}$	40.84$_{(\downarrow 6.1)}$	34.15$_{(\uparrow 12.5)}$	50.38$_{(\uparrow 8.7)}$
GPT-4o-mini	Full	33.62$_{(-)}$	54.17$_{(-)}$	49.42$_{(-)}$	39.29$_{(-)}$	55.37$_{(-)}$
	RAG	29.68$_{(\downarrow 11.7)}$	53.27$_{(\downarrow 1.7)}$	50.51$_{(\uparrow 2.2)}$	39.69$_{(\uparrow 1.0)}$	61.45$_{(\uparrow 11.0)}$
	CoA	26.20$_{(\downarrow 22.1)}$	45.06$_{(\downarrow 16.8)}$	43.09$_{(\downarrow 12.8)}$	50.10$_{(\uparrow 27.5)}$	70.96$_{(\uparrow 28.2)}$
	MemoryBank	28.68$_{(\downarrow 14.7)}$	50.56$_{(\downarrow 6.7)}$	47.28$_{(\downarrow 4.3)}$	40.68$_{(\uparrow 3.5)}$	56.39$_{(\uparrow 1.8)}$

Table 2: Performance comparison of different long context methods across datasets using four base models. Numbers in parentheses represent relative performance differences (in %) relative to Full-Context (Full), where ↑ indicates improvements and ↓ indicates degradation.

methods as well. This indicates that the effectiveness of long context methods is independent of models' intrinsic capabilities.

Given that the effectiveness of long context methods is independent of the foundation model, we average the relative performance of long context methods across the four foundation models to clearly demonstrate their stable relative performance, as shown in Figure 3. We observe that long context methods yield performance improvements on 2WikiMQA and MuSiQue, whereas they result in performance degradation on NarrativeQA, MultiFieldQA, and Qasper.



(a) DCDS (b) Isolated Chunk Ratio

Figure 4: (a) Average DCDS on different datasets. (b) Average proportion of Isolated Chunks on different datasets.

5 Quantifying Context Dependency in Long Context Documents

Given that decomposition (e.g., chunk-wise iterative processing, chunk retrieval) is a fundamental operation in long context methods, we hypothesize that this operation disrupts the context dependency structure inherent in long documents. For documents with strong context dependency, such decomposition-based approaches cause long context methods to fail at modeling this structure, resulting in significant performance degradation. In contrast, Full-Context preserves the complete context dependency structure, allowing it to effectively model these dependencies. For documents with weak context dependency, however, these long context methods are not affected by the disruption of the context structure, allowing them to achieve superior performance compared to Full-Context. To validate this hypothesis, we first propose a metric to quantify the strength of context dependency in a document, and we then measure the context dependency scores for our five evaluation datasets.

5.1 Metric Definition

To quantify the strength of context dependency of a long document, we propose a metric based on inter-chunk dependency judgments. The core idea

is that a document exhibits strong context dependency if understanding a given chunk frequently requires information from preceding chunks. To implement this, we first partition the document D into N sequential chunks.

To comprehensively quantify context dependency across multiple dimensions, and to ensure consistency between the LLM evaluator and human annotators as well as the reproducibility of the evaluation process, we decompose the abstract notion of context dependency into three concrete, operational categories:

- **Information Dependency**: The correct interpretation of c_i requires specific information (e.g., entity definitions or contextual background) explicitly introduced in c_j.
- **Referential Dependency**: c_i contains referential expressions (e.g., pronouns or definite noun phrases) whose antecedents are established in c_j.
- **Logical Dependency**: The assertion or conclusion in c_i is logically derived from premises established in c_j.

Based on these three categories, we define a pairwise dependency evaluator $\mathcal{E}(c_i, c_j)$, which outputs 1 if c_i exhibits any of the above dependency types with respect to c_j, and 0 otherwise. Detailed examples for each dependency type are provided in Appendix H.

Based on these pairwise evaluations, the Chunk Context Dependency Score $S(c_i)$ for chunk c_i is defined as the average of the evaluator outputs between c_i and its k preceding chunks:

$$S(c_i) = \frac{1}{k} \sum_{j=1}^{k} \mathcal{E}(c_i, c_{i-j})$$

Finally, the Document Context Dependency Score (DCDS) for the entire document D is computed as the average of all applicable chunk scores:

$$DCDS(D) = \frac{1}{N-k} \sum_{i=k+1}^{N} S(c_i)$$

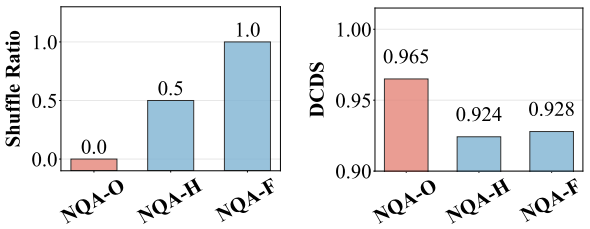
A higher DCDS signifies stronger context dependency. In our implementation, we segment documents into chunks of 512 words, set the window size $k = 3$, and employ GPT-4O-MINI-2024-07-18 as the evaluator. To ensure reliability, we validated this LLM-based approach against human annotations, achieving strong agreement (detailed in Appendix D). The specific prompt template used for evaluation is provided in Appendix C.

5.2 Results and Analysis

We present the average DCDS and the average Isolated Chunk Ratio (defined as the proportion of chunks with a context dependency score of 0) calculated for each dataset. As shown in Figure 4, datasets such as NarrativeQA, MultiFieldQA, and Qasper exhibit high average DCDS coupled with a low proportion of isolated chunks, indicating the presence of dense context dependency structures. Conversely, MuSiQue and 2WikiMQA display significantly lower DCDS and a higher ratio of isolated chunks, suggesting sparse context dependency structures. These empirical findings align well with the relative performance trends observed in Section 4, validating our hypothesis: long context methods are prone to failure on documents with strong context dependency, whereas they are more likely to yield performance gains over Full-Context on documents with weak context dependency.

6 Analyzing the Causes of Long Context Method Failure

To validate the hypothesis that the failure of long context methods stems from their decomposition mechanisms' inability to effectively utilize context dependency information, we designed a controlled experiment to disrupt the context dependency structure of documents. Our core rationale rests on the distinct sensitivities of the Full-Context baseline and long context methods to structural disruption: Full-Context relies heavily on the intact context dependency structure and is thus highly sensitive to its disruption. In contrast, long context methods, constrained by their inherent decomposition, fundamentally struggle to utilize this structure, rendering them relatively insensitive. Based on this asymmetry, we predict that as the dependency structure in documents with strong context dependency is disrupted, the performance degradation of Full-Context will disproportionately exceed that of long context methods, resulting in a narrowing of the performance gap, which manifests as an upward trend in the latter's relative performance. Consequently, observing such an improvement in relative performance under structural disruption serves as evidence of long context methods' inability to effectively utilize context dependency structures.



(a) Applied Shuffle Ratios (b) DCDS across datasets

Figure 5: (a) Applied shuffle ratios for the three datasets. (b) DCDS across the three datasets.

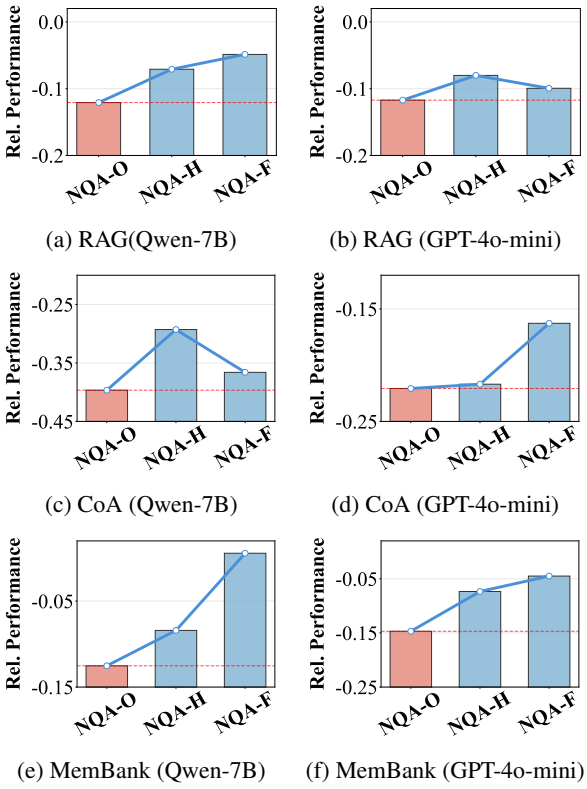
6.1 Experimental Setup

We selected the NarrativeQA dataset, which exhibits strong context dependency, and denote its original version as NQA-O. Using our progressive shuffling algorithm (Appendix, Algorithm 1), we generated NQA-H by shuffling 50% of its chunks, and NQA-F by shuffling all chunks. On these three dataset variants, we measured the relative performance of RAG, CoA, and MemoryBank against Full-Context, using two distinct base models.

6.2 Results and Analysis

First, we verify the effectiveness of the structural disruption. With the applied shuffle ratios illustrated in Figure 5 (a), Figure 5 (b) shows that both NQA-H and NQA-F exhibit significantly lower Document Context Dependency Scores (DCDS) compared to NQA-O. Interestingly, the DCDS of NQA-H and NQA-F are similar. This is likely because further shuffling, while disrupting the context dependencies for some chunk pairs, simultaneously restored the context dependencies for others.

Building on this, the relative performance shifts in Figure 6 demonstrate that long context methods exhibit higher relative performance on datasets with disrupted context dependency structures compared to those with intact structures. This result validates our hypothesis. Specifically, this phenomenon highlights the divergence in response to structural disruption: Full-Context suffers severe performance degradation due to its heavy reliance on the intact dependency structure, whereas long context methods remain relatively unaffected as they inherently struggle to utilize such structure. Ultimately, this increase in relative performance, driven by the decline of Full-Context, indicates that long context methods struggle to effectively model and utilize context dependency structures, rendering them prone to failure on documents with strong context dependency.



(a) RAG(Qwen-7B) (b) RAG (GPT-4o-mini)
(c) CoA (Qwen-7B) (d) CoA (GPT-4o-mini)
(e) MemBank (Qwen-7B) (f) MemBank (GPT-4o-mini)

Figure 6: Relative performance of RAG, CoA, and MemoryBank(MemBank) on NarrativeQA variants with varying levels of shuffling.

7 CoDaR: Context Dependency-Aware Adaptive Routing

Our research indicates that the Vanilla strategy (which uniformly applies long context methods to all inputs) is fragile. It faces a significant risk of performance degradation when processing documents with strong context dependency, compared to Full-Context. To address this, we propose an adaptive strategy, CoDaR (**Co**ntext **d**ependency-**a**ware **r**outer), designed to adaptively invoke these methods only for suitable documents based on the strength of the document's context dependency.

CoDaR Our strategy routes only documents with weak context dependency to the long context method; all other inputs default to Full-Context. Specifically, CoDaR operates based on the DCDS: it invokes the long context method when the document's DCDS falls below a threshold τ; otherwise, it defaults to Full-Context.

7.1 Experimental Setup

For each dataset, we randomly select 20 samples as a validation set to determine the optimal threshold τ for CoDaR, with the remaining samples

Method	Strategy	NarrativeQA		Qasper		MultiFieldQA		MuSiQue		2WikiMQA		AVG	
		F1	Δ%	F1	Δ%	F1	Δ%	F1	Δ%	F1	Δ%	F1	Δ%
RAG	Vanilla	29.19	0.00	**51.06**	0.00	52.46	0.00	40.59	0.00	**61.06**	0.00	46.87	0.00
	Random	30.09	+3.08	50.78	-0.55	53.25	+1.51	40.54	-0.12	59.53	-2.51	46.84	-0.06
	Reflection	30.22	+3.55	50.54	-1.01	53.68	+2.32	40.07	-1.28	58.86	-3.61	46.67	-0.42
	QaR	30.82	+5.60	49.01	-4.01	53.83	+2.61	39.31	-3.14	60.81	-0.40	46.76	-0.24
	CoDaR	**32.50**	+11.34	50.09	-1.90	**53.97**	+2.88	**41.71**	+2.76	60.69	-0.61	**47.79**	+1.96
CoA	Vanilla	26.80	0.00	43.95	0.00	43.73	0.00	**50.92**	0.00	**72.49**	0.00	47.58	0.00
	Random	28.37	+5.86	47.16	+7.30	47.65	+8.96	40.50	-20.46	63.93	-11.81	45.52	-4.33
	Reflection	29.54	+10.25	47.86	+8.91	**53.26**	+21.80	47.87	-5.98	65.70	-9.37	48.85	+2.67
	QaR	29.21	+9.00	48.28	+9.86	51.13	+16.93	49.88	-2.03	67.80	-6.47	49.26	+3.54
	CoDaR	**32.71**	+22.05	**49.33**	+12.24	47.09	+7.68	49.99	-1.83	70.91	-2.18	**50.01**	+5.11
MemBank	Vanilla	27.96	0.00	48.51	0.00	49.50	0.00	41.38	0.00	56.31	0.00	44.73	0.00
	Random	30.49	+9.05	49.69	+2.43	50.93	+2.89	40.48	-2.17	57.16	+1.51	45.75	+2.28
	Reflection	31.89	+14.07	49.78	+2.63	53.00	+7.06	41.54	+0.38	55.59	-1.28	46.36	+3.64
	QaR	29.06	+3.96	48.31	-0.41	52.61	+6.27	41.02	-0.87	**58.65**	+4.17	45.93	+2.68
	CoDaR	**32.60**	+16.60	**50.08**	+3.24	**53.88**	+8.85	**41.59**	+0.51	55.01	-2.31	**46.63**	+4.25
ReadAgent	Vanilla	32.96	0.00	49.53	0.00	50.38	0.00	41.29	0.00	51.88	0.00	45.21	0.00
	Random	32.94	-0.06	49.55	+0.04	52.00	+3.22	**41.83**	+1.31	**55.69**	+7.34	**46.41**	+2.65
	Reflection	32.66	-0.90	49.24	-0.58	52.94	+5.08	40.98	-0.75	53.12	+2.40	45.79	+1.29
	QaR	32.07	-2.69	49.52	-0.03	51.33	+1.89	41.44	+0.37	51.08	-1.55	45.09	-0.27
	CoDaR	**32.97**	+0.03	**50.04**	+1.03	**54.00**	+7.19	40.80	-1.19	51.81	-0.13	45.92	+1.57

Table 3: Performance comparison of different routing strategies across four long context methods. MemBank denotes MemoryBank. We report F1 scores and the relative performance change (Δ%) relative to the Vanilla baseline. The best results within each block are bolded.

used for testing (see Appendix G.1 for an analysis of how validation set size affects routing performance). All reported results are on these test samples. We compare our strategy against four baselines: (1) Vanilla (uniformly applying the long context method); (2) Random (randomly selecting between Full-Context and the long context method with 50% probability); (3) Question-aware Router (QaR), which we construct following the approach in (Aurelio, 2025) (prompting the model to classify the question type as Factoid or Analytical, routing Factoid questions to the long context method and Analytical questions to Full-Context. This baseline represents the effectiveness of routing strategies based on question-level features); and (4) Reflection (Li et al., 2024d) (a post-hoc strategy based on model self-evaluation. It initially generates an answer using Full-Context; if the output is assessed as low quality, the input is then routed to the long context method for re-generation).

7.2 Results and Analysis

As shown in Table 3, CoDaR significantly enhances the robustness of long context methods. On datasets with strong context dependency (NarrativeQA, Qasper, and MultiFieldQA), CoDaR effectively improves performance compared to Vanilla. Notably, on NarrativeQA, CoDaR yields relative improvements of +22.05% for CoA and +16.60% for MemoryBank. On datasets with weak context dependency (MuSiQue and 2WikiMQA), CoDaR yields performance comparable to Vanilla. The slight decline of CoDaR relative to the Vanilla strategy on these datasets is understandable, as long context methods inherently outperform Full-Context on datasets with weak context dependency. On these datasets, the goal of routing between Full-Context and long context methods is to minimize the performance degradation caused by Full-Context. As evidenced in Table 3, CoDaR preserves the advantages of long context methods on these documents more effectively than all other routing baselines.

Furthermore, CoDaR demonstrates superior reliability, largely outperforming both the Question-aware Router (QaR) and Reflection strategies. This performance advantage validates the effectiveness of DCDS as a routing signal, highlighting the necessity of analyzing document-level features. Although the Random strategy occasionally surpasses

CoDaR in specific instances (e.g., with ReadAgent on MuSiQue and 2WikiMQA), it exhibits extreme instability overall. Further detailed analysis regarding the unreliability of the Random strategy is presented in Appendix G.3. To further demonstrate the generalizability of CoDaR, we also evaluate its performance on summarization tasks in Appendix G.2, where it significantly outperforms both the Vanilla and Random strategies.

8 Conclusion

In this paper, we systematically investigate the effectiveness of long context methods across different types of datasets. We find that long context methods are not universally effective, and indiscriminately deploying them could pose significant performance risks. Furthermore, we investigated the root causes of the failure of long context methods and find that their intrinsic decomposition mechanisms hinder context dependency modeling, causing these methods to suffer performance declines on documents with strong context dependency. Finally, to enhance the robustness of long context methods across different datasets, we propose CoDaR, a training-free adaptive strategy that effectively mitigates the performance degradation on documents with strong context dependency while preserving the advantages of these methods on documents with weak context dependency.

9 Limitations

While our work demonstrates that long context methods struggle to effectively model context dependency due to decomposition mechanism, we have not conducted an in-depth analysis of the impact of decomposition Mechanism from the perspective of attention mechanisms. Additionally, our evaluation is exclusively based on English datasets, and the generalizability to languages with distinct discourse structures and syntactic dependencies warrants further verification.

10 Ethical Considerations

We strictly adhere to ethical standards to ensure the integrity and positive impact of our research. We exclusively utilize established open-source datasets (e.g., LongBench) and recognized models, ensuring our work relies on accessible resources free of personally identifiable information (PII). Regarding the human evaluation detailed in Appendix D, we followed strict ethical guidelines; all annotators were fully informed and compensated at rates exceeding local minimum wage standards. Furthermore, our proposed framework allows for more reliable long context processing, mitigating the risks of generating misleading information. Additionally, AI tools were used for language polishing of this paper.

11 Acknowledgements

This work was funded by the New Generation Artificial Intelligence-National Science and Technology Major Project 2025ZD0123301,the Strategic Priority Research Program of the CAS under Grants No. XDB0680102, the National Natural Science Foundation of China (NSFC) under Grants No. 62441229.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, S'l'bastien Bubeck, Martin Cai, Caio C'l'sar Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 66 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. Technical Report MSR-TR-2024-12, Microsoft.

Anthropic. 2025. System card: Claude sonnet 4.5. Technical report, Anthropic. Accessed: 2025-10-23.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection.

Aurelio. 2025. Semantic router: A superfast decision-making layer for llms. https://github.com/aurelio-labs/semantic-router.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, and 1 others. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Adithya Bhaskar, Alexander Wettig, Tianyu Gao, Yihe Dong, and Danqi Chen. 2025. Cache me if you can: How many kvs do you need for effective long-context lms? *arXiv preprint arXiv:2506.17121*.

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv preprint arXiv:2404.00610*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. 2025. How to train long-context language models (effectively). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7376–7399.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Jinwu Hu, Wei Zhang, Yufeng Wang, Yu Hu, Bin Xiao, Mingkui Tan, and Qing Du. 2025. Dynamic compressing prompts for efficient inference of large language models. *arXiv preprint arXiv:2504.11004*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H Abdi, Dongsheng Li, Chin-Yew Lin, and 1 others. 2024a. Minference 1.0: Accelerating prefilling for long-context llms via dynamic sparse attention. *Advances in Neural Information Processing Systems*, 37:52481–52515.

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

Ziyan Jiang, Xueguang Ma, and Wenhu Chen. 2024c. Longrag: Enhancing retrieval-augmented generation with long-context llms. *arXiv preprint arXiv:2406.15319*.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Alex Laitenberger, Christopher D Manning, and Nelson F Liu. 2025. Stronger baselines for retrieval-augmented generation with long-context language models. *arXiv preprint arXiv:2506.03989*.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. 2024. A human-inspired reading agent with gist memory of very long contexts. *arXiv preprint arXiv:2402.09727*.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2024a. Loogle: Can long-context language models understand long contexts? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16304–16333.

Siheng Li, Cheng Yang, Zesen Cheng, Lemao Liu, Mo Yu, Yujiu Yang, and Wai Lam. 2024b. Large language models can self-improve in long-context reasoning. *arXiv preprint arXiv:2411.08147*.

Yanyang Li, Shuo Liang, Michael Lyu, and Liwei Wang. 2024c. Making long-context language models better multi-hop reasoners. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2462–2475.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024d. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. *arXiv preprint arXiv:2407.16833*.

Jiaheng Liu, Dawei Zhu, Zhiqi Bai, Yancheng He, Huanxuan Liao, Haoran Que, Zekun Wang, Chenchen Zhang, Ge Zhang, Jiebin Zhang, and 1 others. 2025. A comprehensive survey on long context language modeling. *arXiv preprint arXiv:2503.17407*.

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

Lingrui Mei, Jiayu Yao, Yuyao Ge, Yiwei Wang, Baolong Bi, Yujun Cai, Jiazhi Liu, Mingyu Li, Zhong-Zhi Li, Duzhen Zhang, and 1 others. 2025. A survey of context engineering for large language models. *arXiv preprint arXiv:2507.13334*.

Jacob Menick, Kevin Lu, Shengjia Zhao, E Wallace, H Ren, H Hu, N Stathas, and F Petroski Such. 2024. Gpt-4o mini: advancing cost-efficient intelligence. *Open AI: San Francisco, CA, USA*.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. 2025. Memorag: Boosting long context processing with global memory-enhanced retrieval augmentation. In *Proceedings of the ACM on Web Conference 2025*, pages 2366–2377.

Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, and Zhicheng Dou. 2024. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *arXiv preprint arXiv:2409.05591*, 1.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Kaiqiang Song, Xiaoyang Wang, Sangwoo Cho, Xiaoman Pan, and Dong Yu. 2023. Zebra: Extending context window with layerwise grouped local-global attention. *arXiv preprint arXiv:2312.08618*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. ‘Ź‘ń musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, and 1 others. 2024. Memoryllm: Towards self-updatable large language models. *arXiv preprint arXiv:2402.04624*.

Jeffrey Willette, Heejun Lee, Youngwan Lee, Myeongjae Jeon, and Sung Ju Hwang. 2024. Training-free exponential context extension via cascading kv cache. *arXiv preprint arXiv:2406.17808*.

Longyun Wu, Dawei Zhu, Guangxiang Zhao, Zhuocheng Yu, Junfeng Ran, Xiangyu Wong, Lin Sun, and Sujian Li. 2025. Longattn: Selecting long-context training data via token-level attention. *arXiv preprint arXiv:2502.16860*.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Retrieval meets long context large language models. *arXiv preprint arXiv:2310.03025*.

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

Xi Ye, Fangcong Yin, Yinghui He, Joie Zhang, Howard Yen, Tianyu Gao, Greg Durrett, and Danqi Chen. 2025. Longproc: Benchmarking long-context language models on long procedural generation. *arXiv preprint arXiv:2501.05414*.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, and 1 others. 2025. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. *arXiv preprint arXiv:2507.02259*.

Wenhao Yu, Hongming Zhang, Xiaoman Pan, Peixin Cao, Kaixin Ma, Jian Li, Hongwei Wang, and Dong Yu. 2024. Chain-of-note: Enhancing robustness in retrieval-augmented language models. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 14672–14685.

Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, and 1 others. 2024. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k. *arXiv preprint arXiv:2402.05136*.

Zhenrui Yue, Honglei Zhuang, Aijun Bai, Kai Hui, Rolf Jagerman, Hansi Zeng, Zhen Qin, Dong Wang, Xuanhui Wang, and Michael Bendersky. 2024. Inference scaling for long-context retrieval augmented generation. *arXiv preprint arXiv:2410.04343*.

Jiajie Zhang, Zhongni Hou, Xin Lv, Shulin Cao, Zhenyu Hou, Yilin Niu, Lei Hou, Yuxiao Dong, Ling Feng, and Juanzi Li. 2025. Longreward: Improving long-context large language models with ai feedback. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3718–3739.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2024a. Soaring from 4k to 400k: Extending llm‘Ă‘Źs context with activation beacon. *arXiv preprint arXiv:2401.03462*, 2(3):5.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Arik. 2024b. Chain of agents: Large language models collaborating on long-context tasks. *Advances in Neural Information Processing Systems*, 37:132208–132237.

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

Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, Yuxiao Dong, and Jie Tang. 2024b. Longrag: A dual-perspective retrieval-augmented generation paradigm for long-context question answering. *arXiv preprint arXiv:2410.18050*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and 1 others. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19724–19731.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Peng Cui, Tiannan Wang, Zhenxin Xiao, Yifan Hou, Ryan Cotterell, and Mrinmaya Sachan. 2023. Recurrentgpt: Interactive generation of (arbitrarily) long text. *arXiv preprint arXiv:2305.13304*.

A Detailed Method Descriptions and Experimental Configurations

This appendix provides comprehensive descriptions of the four representative long context methods evaluated in our study, along with their specific experimental configurations to ensure reproducibility.

A.1 Naive RAG

Method Description Naive RAG (Xu et al., 2023) addresses long-context processing by retrieving and processing only relevant document segments rather than entire documents, thereby enhancing the model's focus on pertinent information.

Formally, given a document D and question q, RAG first segments D into chunks $\{c_1, \ldots, c_n\}$. For each chunk, a dense retriever computes embedding representations and measures relevance to the question via cosine similarity. The top-k most relevant chunks $\{c_{i_1}, \ldots, c_{i_k}\}$ are then concatenated and fed to the language model for answer generation:

$$W_{\text{RAG}}(D, q) = M(\text{concat}(c_{i_1}, \ldots, c_{i_k}), q), \quad (4)$$

where M denotes the language model and chunks are ordered by descending similarity scores.

Experimental Setup We employ BGE-M3 (Chen et al., 2024), a multilingual dense retriever with 1024-dimensional embeddings. Documents are partitioned into 512-token chunks using `semantic-text-splitter`. For each question, we retrieve the top-15 chunks ($k = 15$) ranked by cosine similarity and concatenate them in descending order. For generation, we follow the prompt templates and maximum token limits specified in LongBench (Bai et al., 2023).

A.2 Chain of Agents (CoA)

Method Description Chain of Agents (CoA) (Zhang et al., 2024b) employs a multi-agent collaboration paradigm to distribute long-context processing tasks across multiple agents, thereby alleviating the attention burden on individual models.

The workflow operates as follows: Given a document D and question q, the document is partitioned into m sequential chunks $\{c_1, \ldots, c_m\}$. Worker agents M_{worker} sequentially process their assigned chunks, with each worker agent i receiving its chunk c_i, the previous agent's output o_{i-1}, and the question q:

$$o_i = M_{\text{worker}}(c_i, o_{i-1}, q), \quad i = 1, \ldots, m \quad (5)$$

After all chunks are processed, a manager agent M_{manager} synthesizes the final worker output o_m with the question to generate the answer a:

$$a = M_{\text{manager}}(o_m, q) \quad (6)$$

CoA processes the entire input by interleaving reading and reasoning, and it mitigates long context focus issues by assigning each agent a short context.

Experimental Setup Following (Zhang et al., 2024b), we partition documents into 8,192-word segments and use the same language model for both worker and manager agents. Worker agents sequentially process chunks and pass their complete outputs to the next agent, while the manager receives only the final worker output and question to generate the answer. For generation, we follow the prompt templates and maximum token limits specified in LongBench (Bai et al., 2023).

A.3 MemoryBank

Method Description MemoryBank (Zhong et al., 2024) draws inspiration from human cognitive processes of reading long texts by constructing hierarchical memory structures that retain important information while discarding redundancies.

The method processes chunked documents sequentially, maintaining a multi-layer memory state $\mathcal{M}_t$ at each step t. When processing chunk c_t, MemoryBank updates its memory by integrating new information with previous memory states:

$$\mathcal{M}_t = \text{Update}(\mathcal{M}_{t-1}, c_t), \quad t = 1, \ldots, n \quad (7)$$

MemoryBank constructs hierarchical memory through summarization, enabling the model to access both detailed and summarized representations. After processing all n chunks, the answer is generated by retrieving relevant information from the constructed memory $\mathcal{M}_n$ based on the question:

$$a = M(\text{Retrieve}(\mathcal{M}_n, q), q) \quad (8)$$

Experimental Setup In our experiments, we segment the document into 512-token chunks. We utilize BGE-M3 (Chen et al., 2024) to retrieve the top-15 memory chunks. These retrieved chunks are arranged in descending order of similarity to

construct a query-relevant memory context, which is subsequently used to generate the answer. For the generation phase, we adhere to the prompt templates and maximum token limits specified in LongBench (Bai et al., 2023).

A.4 ReadAgent

Method Description ReadAgent (Lee et al., 2024) employs a compression-based approach that mimics human reading strategies by first skimming through content to identify relevant sections, then performing detailed reading of those sections.

The method operates in three stages. First, the raw text is segmented into coherent pages $\{p_1, \ldots, p_l\}$ via Episode Pagination, where the model identifies natural semantic breaks to determine boundaries. Second, each page p_i is compressed into a concise gist memory s_i that captures key information:

$$s_i = \text{Compress}(p_i), \quad i = 1, \ldots, l, \tag{9}$$

where l is the total number of pages. Third, given the question q, the model uses these gist memories to identify the most relevant pages $\{p_{j_1}, \ldots, p_{j_r}\}$:

$$\{p_{j_1}, \ldots, p_{j_r}\} = \text{Retrieve}(\{s_1, \ldots, s_l\}, q) \tag{10}$$

Finally, the full content of the retrieved pages is concatenated and provided to the model for detailed answer generation:

$$a = M(\text{concat}(p_{j_1}, \ldots, p_{j_r}), q) \tag{11}$$

This hierarchical strategy significantly reduces the initial context processing burden while maintaining access to detailed information when needed.

Experimental Setup Following (Lee et al., 2024), we set max_words=1,200 to define the maximum page length during episode pagination and configure the model to retrieve up to 3 pages ($r \leq 3$) for answer generation. For generation, we follow the prompt templates and maximum token limits specified in LongBench (Bai et al., 2023).

B Dataset Statistics

The statistics of the datasets we used are shown in Table 4.

Dataset	#Samples	Avg Len	Max Len
NarrativeQA	200	29,837	65,301
MultiFieldQA	150	7,119	16,446
Qasper	200	5,026	21,879
MuSiQue	200	16,269	17,824
2WikiMQA	200	7,475	16,982

Table 4: Dataset statistics. The token number is calculated using the Qwen2.5 tokenizer. #Samples denote the total number of benchmarks.

C Pairwise Dependency Evaluator

The prompt used to evaluate whether text block B exhibits context dependency on the preceding text block A is as follows:

Prompt Template for Dependency Evaluator

Determine whether text block B exhibits context dependency on text block A. Answer "yes" if comprehending B requires information, discourse state, or logical premises established in A. Answer "no" if B can be interpreted independently, introduces distinct entities, or lacks substantive connection to A.

Text Block A: `{text_block_a}`
Text Block B: `{text_block_b}`
Answer:

D Human Annotation and Evaluator Validation

To validate the reliability of our LLM-based pairwise dependency evaluator, we constructed a human-annotated dataset and measured its agreement against human judgments.

D.1 Annotation Guidelines

Annotators were instructed to evaluate whether a subsequent chunk B exhibited context dependency on its preceding chunk A. The task required selecting one of two labels:

- YES : Understanding the semantic content of chunk B requires information, discourse states, or logical premises established in chunk A.
- NO : Chunk B can be understood independently, introduces different entities, or lacks a substantive connection to chunk A.

We primarily considered the following three types of context dependency:

- Information Dependency: The correct interpretation of chunk B requires specific information (e.g., entities, concept definitions, or contextual background) explicitly introduced in A.
- Referential Dependency: Chunk B contains explicit referential expressions (e.g., pronouns like 'he'/'it', demonstratives like 'this'/'that', or definite noun phrases) that are anaphoric, requiring

an antecedent (an entity or topic) established in A for their correct resolution.

- Logical Dependency: The assertion, argument, or conclusion in B is logically derived from, or built upon, premises or findings established in A.

D.2 Annotation Process

Data Sampling We randomly sampled 20 documents from each of the five datasets (Qasper, NarrativeQA, MultiFieldQA, 2WikiMQA, and MuSiQue) and segmented them into 512-word chunks. We then paired each chunk with its k preceding chunks (where $k = 3$, consistent with the DCDS metric) and randomly selected 600 pairs in total for annotation.

Two-Stage Annotation The annotation was conducted in two stages:

1. Pilot Study: Three annotators independently labeled 100 samples to assess and calibrate the annotation guidelines. We calculated a Fleiss' Kappa coefficient of 0.75, indicating substantial agreement.

2. Formal Annotation: The remaining 500 samples were annotated using a "dual-annotation plus adjudication" mechanism. Each sample was first labeled by two annotators. If they agreed, the label was accepted. If they disagreed, a third senior annotator served as an adjudicator, and the final "Gold Standard" label was determined by majority vote.

D.3 Pairwise Dependency Evaluator Reliability

To verify the reliability of the pairwise dependency evaluator used in this paper (implemented via GPT-4O-MINI-2024-07-18), we compared its automated annotations against the human-annotated "Gold Standard" (derived from the majority vote in the formal annotation stage).

The results show that the pairwise dependency evaluator achieved an 84.6% accuracy compared to human judgments. The Cohen's Kappa coefficient between the evaluator and the gold standard was 0.64. This Kappa value signifies substantial agreement, confirming that the pairwise dependency evaluator provides context dependency assessments that are highly consistent with human judgment and supporting its application for large-scale evaluation in our study.

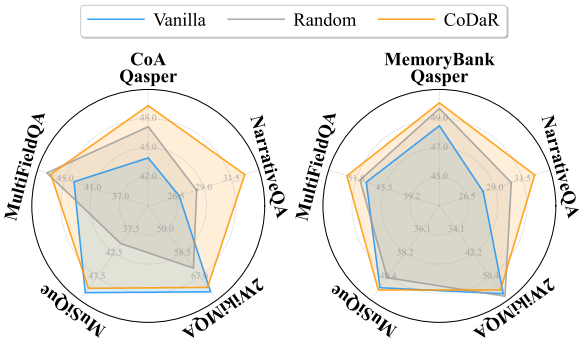


Figure 7: F1 performance of long context methods (CoA and MemoryBank) under three strategies.

E Prompt for QaR and Reflection

The prompt template used in the QaR strategy to analyze the question type is provided below.

QaR prompt template that we use for our baseline

Classify the following question into Factoid or Analytical.

Factoid: Seeks specific entities, dates, numbers, or concrete details.
Analytical: Requires reasoning, summarization, inferring causality, or comprehensive understanding.

Question: `{question}`
Category:

The prompt template used in the Reflection strategy to self-evaluate the quality of the Full-Context answer is provided below.

Prompt Template for Self-Evaluating Answer Quality

You are a strict answer evaluator. Your task is to determine if the "Answer" is a correct and complete response to the "Question," based only on the provided "Context."

- If the Answer is fully correct, complete, and derived entirely from the Context, respond only with "yes".
- If you determine the Answer is incorrect, incomplete, or contains information not found in the Context (i.e., it is a hallucination), which means an alternative method is needed to solve the problem, respond only with "no".

Context:
`{context}`

Question: `{question}`
Answer to Evaluate: `{answer}`
Your Evaluation:

F Discussion on the Computational Overhead of CoDaR

This section discusses the overhead introduced by the calculation of the Document Context Dependency Score (DCDS) within the CoDaR strategy. Although the DCDS calculation involves traversing document chunks and invoking LLMs, we argue

that CoDaR remains efficient and feasible in long-context scenarios for the following two reasons:

- **One-time Computation and Amortization:** DCDS is a query-independent intrinsic attribute of the document, requiring calculation only once during the preprocessing phase. In typical multi-turn QA scenarios, as the number of queries N increases, the calculation time is amortized, causing the additional overhead per query to approach zero. Thus, calculating the document's DCDS in the CoDaR strategy is highly cost-effective.

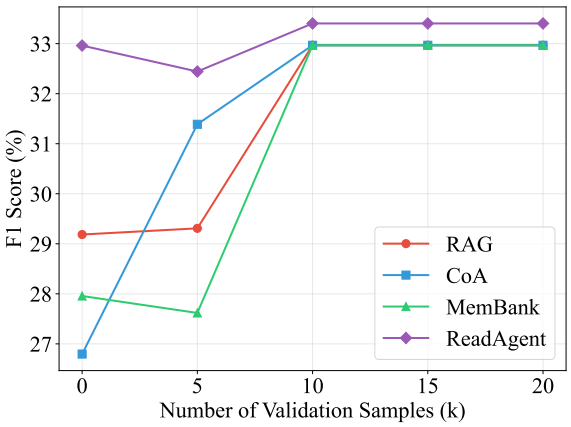
- **High Parallelizability:** The calculation of DCDS is based on a sliding window mechanism, where the dependency judgments between chunks are logically independent. Therefore, DCDS calculation supports parallel computing, which ensures rapid completion in engineering implementations without imposing a significant impact on overall system latency.

G Supplementary Experiments

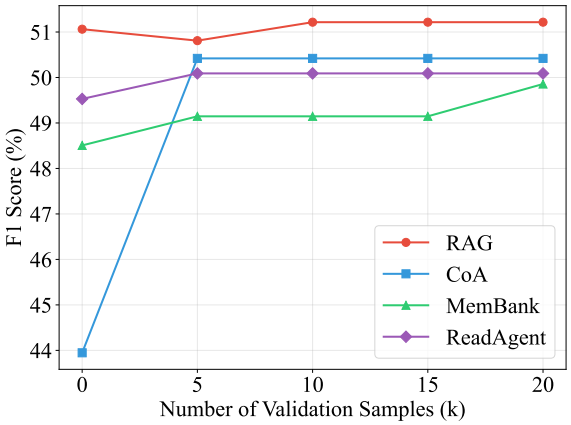
G.1 Impact of Validation Set Size on Routing Performance

To evaluate the impact of validation set size on the routing performance of CoDaR, we conducted ablation experiments on the NarrativeQA and Qasper datasets. In this setup, we utilized the same test set as employed in our main evaluations and randomly sampled k instances from the remaining data pool to serve as a validation set for determining the optimal threshold τ.

Our experimental results, illustrated in Figure 8, indicate that as the number of validation samples increases, the performance of long context methods on the test set rises progressively and reaches a saturation point at $k = 10$. This finding indicates that a small number of samples is sufficient to determine a suitable routing threshold. Moreover, after reaching the saturation point, the F1 scores exhibit high stability even as the validation sample size increases further. Consequently, these results suggest that selecting 20 samples for threshold determination in our main experiments is reasonable and effectively represents the true performance of the CoDaR strategy.



(a) NarrativeQA



(b) Qasper

Figure 8: Impact of the number of validation samples (k) on the F1 performance of different long context methods. The performance reaches a saturation point at $k = 10$ across different datasets.

G.2 Cross-Task Generalization to Summarization

We evaluate the performance of CoDaR on long context summarization tasks. Specifically, we utilize two representative datasets sourced from LongBench (Bai et al., 2023): GovReport (Huang et al., 2021), which consists of long-form government reports, and QMSum (Zhong et al., 2021), which focuses on query-based meeting summarization. By dynamically identifying the context dependency strength of documents, CoDaR effectively mitigates the risk of information loss inherent in decomposition mechanisms. As illustrated in Table 5, CoDaR significantly outperforms both the Vanilla and Random baselines across all ROUGE metrics. Notably, on the QMSum dataset, CoDaR achieves a substantial 53.01% improvement in the ROUGE-2 metric for the COA method. This result provides

strong evidence that the effectiveness of CoDaR is not restricted to question-answering tasks but can also seamlessly generalize to summarization tasks, demonstrating robust performance.

G.3 Detailed Analysis of the Unreliability of the Random Strategy

We present a more detailed analysis regarding the unreliability of the Random strategy. As illustrated in Figure 7, when processing datasets with strong context dependency (e.g., NarrativeQA, MultiFieldQA, and Qasper), the Random strategy yields only limited improvements over the Vanilla baseline, whereas CoDaR achieves significantly more substantial gains.

Furthermore, on datasets with weak context dependency (such as MuSiQue and 2WikiMQA), where long context methods typically demonstrate a significant advantage, the Random strategy suffers from a marked performance decline relative to the Vanilla baseline. In contrast, the CoDaR strategy achieves performance comparable to Vanilla.

Notably, the Random strategy is susceptible to "catastrophic failure." As shown in Table 3, applying the Random strategy to CoA on the MuSiQue dataset results in a 20.46% decline in performance relative to the Vanilla baseline. This collapse stems from the fact that the random strategy completely ignores the document's intrinsic dependency structure, failing to adaptively route long context methods based on specific document characteristics.

In summary, the preceding analysis demonstrates that the stability of the Random strategy is significantly inferior to that of CoDaR.

H Detailed Definitions and Examples of Context Dependency

The pairwise dependency evaluator in this study determines whether a text chunk c_i exhibits dependency on a preceding chunk c_j based on three core dimensions: Information Dependency, Referential Dependency, and Logical Dependency. Detailed definitions and typical examples for each dimension are provided below.

- **Information Dependency**
  Definition: The correct interpretation of chunk c_i requires specific information (e.g., entities, concept definitions, or contextual background) explicitly introduced in c_j.

  Example:

  c_j: This paper proposes a new module named "Adaptive Feature Fuser" (AFF) for dynamically adjusting layer weights.

  c_i: Experimental results show that AFF significantly reduces computational overhead when processing multimodal data.

  Analysis: The comprehension of the term "AFF" in c_i relies on the full name and functional definition provided in c_j. Without c_j, the acronym lacks context and cannot be fully understood.

- **Referential Dependency**
  Definition: Chunk c_i contains explicit referential expressions (e.g., pronouns like 'he'/'it', demonstratives like 'this'/'that', or definite noun phrases) that are anaphoric, requiring an antecedent (an entity or topic) established in chunk c_j for their correct resolution.

  Example:

  c_j: The detective examined the crime scene carefully.

  c_i: He found suspicious footprints near the window.

  Analysis: The pronoun "He" in c_i explicitly refers back to "The detective" introduced in c_j. Removing c_j results in the loss of the antecedent for the pronoun.

- **Logical Dependency**
  Definition: The assertion, argument, or conclusion in c_i is logically derived from, or built upon, premises or findings established in c_j.

  Example:

  c_j: Experiments indicate that regular exercise reduces mortality rates by 40%.

  c_i: Therefore, doctors should recommend daily workouts to all patients.

  Analysis: The connective "Therefore" indicates that the recommendation in c_i is a logical consequence of the empirical finding in c_j. The chunk c_i depends on c_j to provide the premise for its argument.

Method	Strategy	gov_report (R-1)		gov_report (R-2)		gov_report (R-L)		qmsum (R-1)		qmsum (R-2)		qmsum (R-L)	
		Value	Δ%	Value	Δ%	Value	Δ%	Value	Δ%	Value	Δ%	Value	Δ%
RAG	Vanilla	33.25	0.00	8.42	0.00	30.13	0.00	28.29	0.00	6.50	0.00	24.21	0.00
	Random	33.90	+1.95	8.70	+3.32	30.65	+1.72	28.52	+0.81	6.35	-2.31	24.66	+1.86
	CoDaR	**34.84**	+4.78	**9.24**	+9.74	**31.58**	+4.81	**29.02**	+2.58	**6.72**	+3.38	**25.06**	+3.51
COA	Vanilla	34.08	0.00	8.80	0.00	30.95	0.00	24.34	0.00	4.15	0.00	20.70	0.00
	Random	34.77	+2.02	9.09	+3.30	31.44	+1.58	26.49	+8.83	5.23	+26.02	22.84	+10.34
	CoDaR	**35.07**	+2.90	**9.39**	+6.70	**31.84**	+2.88	**28.53**	+17.21	**6.35**	+53.01	**24.56**	+18.65

Table 5: Performance comparison of long context methods on summarization tasks. $\Delta\%$ represents the relative improvement over the Vanilla baseline.

I Instance-Level Failure Case Analysis

To further investigate how long context methods fail on documents with strong context dependency, we conduct an instance-level case study using CoA on the NarrativeQA dataset. In documents with strong context dependency, the decomposition mechanism hinders CoA from capturing logical dependencies between chunks, such as narrative plot twists. The information continuously extracted and passed along via the Communication Unit (CU) tends to form a solidified prior: when the plot in a newly processed chunk changes, the CU often fails to link the logical jump back to earlier context, causing its state to remain stale. The resulting error then propagates and accumulates across all subsequent chunks, ultimately producing an incorrect final output.

Table 6 presents a representative failure case from NarrativeQA, illustrating how CoA's CU gets "lost" in a document with strong context dependency.

J Sensitivity of DCDS to Chunk Size

To examine the effect of chunk size on DCDS computation, we randomly sample 20 documents from each of the five datasets and compute the average DCDS under three chunk sizes: 256, 512, and 1024 words. Results are presented in Table 7.

While varying chunk size introduces fluctuations in the absolute DCDS values, the relative ranking of context dependency strength across datasets remains highly consistent (e.g., the score for NarrativeQA is consistently and significantly higher than that of MuSiQue). Within the CoDaR framework, DCDS serves primarily as a relative routing signal. Since the threshold τ is learned dynamically on a validation set, this stable relative ranking guarantees the robustness of the routing strategy, ensuring it remains effective despite variations in chunk size.

K Algorithm

The Progressive Shuffle Algorithm (Algorithm 1) segments the text into fixed-length chunks. Adopting an incremental strategy, it calculates the total number of chunks to be shuffled based on the target ratio, subtracting the already processed ones to determine the new count. While strictly preserving the positions of previously shuffled chunks, it applies permutation solely to the newly selected chunks from the unshuffled set, thereby progressively altering the text sequence.

Question: *Who was the final raid in the story on?*		
Chunk	**Ground Truth Narrative**	**CoA CU Status**
Early Chunk	Rogers and Hearn plan to target the "Bad Bloods".	*Successfully captured:* Summarizes that "Rogers and Hearn devised a plan against the 'Bad Bloods'."
Middle Chunk	*Plot Twist:* The actual target of the raid is the "Sinsings". The previous plan against the "Bad Bloods" was merely a feint to deceive the enemy.	*Dependency Broken:* Fails to capture the logical dependency between chunks. The CU is not updated and retains the outdated "Bad Bloods" raid plan.
Late Chunk	Rogers ultimately leads the real raid on the "Sinsings".	*Error Accumulated:* The erroneous information propagates. The final CU summary solidifies as: "The final raid is on the Bad Bloods."

Table 6: A representative failure case of CoA on NarrativeQA. CoA's final answer is *The Bad Bloods* (incorrect), whereas Full-Context correctly answers *The Sinsings*.

Algorithm 1 Progressive Shuffle Algorithm

Input: Current text chunks $C = \{c_1, c_2, \ldots, c_n\}$, target shuffle ratio $r \in [0, 1]$, set of already shuffled indices S.

Output: Updated chunks C and updated indices set S.

$t \leftarrow \lfloor n \cdot r \rfloor$ {Target number of shuffled chunks}
$\Delta \leftarrow t - |S|$ {Number of new chunks to shuffle}
if $\Delta \leq 0$ **then**
  return C, S {Target ratio already reached}
end if
$U \leftarrow \{i \mid i \in \{1, \ldots, n\}, i \notin S\}$ {Identify unshuffled indices}
$N \leftarrow \text{Sample}(\min(\Delta, |U|), \text{from } U)$ {Select new indices to shuffle}
$S \leftarrow S \cup N$ {Update history of shuffled indices}
if $|N| < 2$ **then**
  return C, S {Need at least 2 new chunks to swap}
end if
$P \leftarrow \{C[i] \mid i \in N\}$ {Extract **only** the new chunks}
$P' \leftarrow \text{Shuffle}(P)$ such that $P'[j] \neq P[j]$ for all j
for each index $i \in N$ **do**
  $C[i] \leftarrow$ next chunk from P' {Update only new positions}
end for
return C, S

Dataset	**256 words**	**512 words**	**1024 words**
NarrativeQA	0.94	0.97	0.99
Qasper	0.94	0.97	0.99
MultiFieldQA	0.88	0.93	0.88
MuSiQue	0.62	0.55	0.42
2WikiMQA	0.43	0.40	0.32

Table 7: Average DCDS across datasets under different chunk sizes.