OkraLong: A Flexible Retrieval-Augmented Framework for Long-Text Question Answering

Anonymous ACL submission

Abstract

Large Language Models (LLMs) encounter challenges in efficiently answering long-text questions, as seen in applications like enterprise document analysis and financial report comprehension. While conventional solutions employ long-context processing or Retrieval-Augmented Generation (RAG), they suffer from prohibitive input expenses or incomplete information. Recent advancements adopt context compression and dynamic retrieval loops, but still sacrifice critical details or incur iterative costs. To address these limitations, we propose OkraLong, a novel framework that flexibly optimizes the entire processing workflow. Unlike prior static or coarse-grained adaptive strategies, OkraLong adopts fine-grained orchestration through three synergistic components: analyzer, organizer and executor. The analyzer characterizes the task states, which guide the organizer in dynamically scheduling the workflow. The executor carries out the execution and generates the final answer. Experimental results demonstrate that OkraLong not only enhances answer accuracy by 5.7%-41.2%, but also achieves cost savings of 1.3x-4.7x.

1 Introduction

Large Language Models (LLMs) have been extensively utilized to handle external knowledge and unseen data, which is a common scenario in real-world applications such as enterprise search and data analysis (Gao et al., 2023; Hui et al., 2024). A critical challenge in these domains, however, lies in querying and comprehending long-form text (Bai et al., 2024). For example, a company may need to query its proprietary technique documents; a financial expert may need to extract insights from the latest corporate reports; and a research group may need to assimilate cutting-edge academic papers to guide their innovations.

To tackle long-text questions, two prevalent methodologies are typically utilized: long-context (LC) and Retrieval-Augmented Generation (RAG) (Li et al., 2024a; Xu et al., 2024b). The LC approach leverages the LLM's inherent ability to process extensive texts by inputting entire content, enabling responses grounded in global contextual awareness (Xu et al., 2024b; Fei et al., 2024). In contrast, RAG employs a lightweight retriever to first identify question-relevant text segments, which are then analyzed by the LLM to get the answer (Lewis et al., 2020; Jeong et al., 2024; Asai et al., 2024).

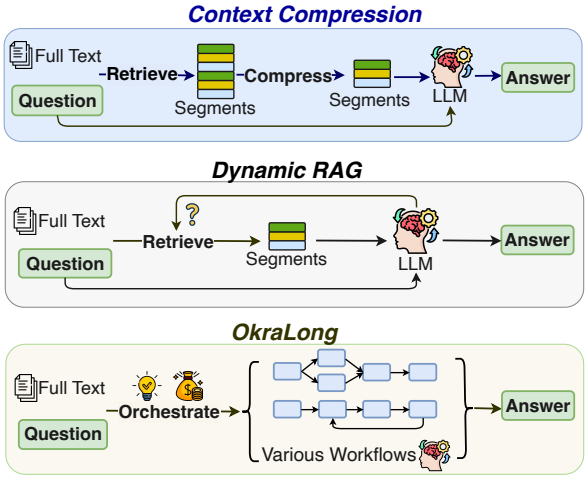

Figure 1: Comparison of OkraLong with two prevalent advanced paradigms for processing long-text questions.

However, when deployed in practical settings, these strategies encounter significant limitations in cost-effectiveness and accuracy. First, given that the cost of LLMs escalate with data volume, employing LC with voluminous text may prove prohibitively expensive (e.g., a financial reports may span hundreds of pages) (Li et al., 2024c; Jiang et al., 2024). While RAG reduces input length by filtering irrelevant text, the context content still incurs moderate costs and risk omitting critical information (Gao et al., 2023). Furthermore, real-world

queries vary widely, from simple fact extraction to multi-step reasoning. Rigid approaches like static retrieval struggle to adapt to this diversity, leading to information loss and inaccuracies (Shao et al., 2023; Zhuang et al., 2024).

Recent efforts to mitigate these limitations primarily focus on two paradigms: context compression and dynamic RAGs. As shown in Figure 1, compression-based approaches typically operate on extensive text segments, removing non-informative tokens or iteratively summarize the content using small generative models (Jiang et al., 2024; Jiang et al., 2023a; Yoon et al., 2024). However, these methods risk losing critical specific details and incur latency overhead due to heavy reliance on small models (Hwang et al., 2024). Dynamic RAG approaches employ iterative retrieval-generation cycles to adaptively make retrieval decisions (Asai et al., 2024; Jiang et al., 2023b; Su et al., 2024). However, iterative workflow requires frequent LLM calls, escalating financial costs, and the existing adaptive mechanisms remain coarse-grained, failing to optimize the performance effectively in varied scenarios.

To address these limitations, we propose OkraLong, a novel retrieval-augmented framework that systematically optimizes long-text question answering. Unlike above approaches that rely on fixed workflow patterns, OkraLong flexibly orchestrates various pipelines according to different task scenarios. As illustrated in Figure 2, our framework comprises three synergistic components: (1) Analyzer: a fine-tuned lightweight model that proactively characterizes task states, utilizing question semantics and preliminary retrieved contexts; (2) Adaptive Organizer: a dynamic scheduler that generates optimized execution plans, based on previous analysis; (3) Executor: a modular operator suite that supports the execution of diverse processing pipelines and strategies.

Distinct from prior adaptive RAG methods (Jiang et al., 2023b; Jeong et al., 2024) that make coarse-grained decisions (e.g. whether to generate iteratively or retrieve additional data) , OkraLong is designed to fine-grainedly optimize the entire processing workflow, covering multiple modules and various pipelines. First, to improve accuracy performance, OkraLong constructs the flexibility to tailor strategies for different tasks. For example, comparative tasks (e.g., Who won the most awards, A, B or C?) demand separate entity retrieval, while multi-step questions (e.g., What is the place of birth of the director of film Clowning Around?) trigger iterative reasoning process. Second, for financial effectiveness, our cost-aware organizer dynamically allocates token budgets and information resources. For instance, general summarizing questions receive multiple aggregated contexts, whereas fact extraction questions utilize targeted context slicing. It is worth noting that these flexible orchestrations are facilitated by the task-understanding analyzer and we also develop several innovative executing operators to support tailored strategies.

We evaluate OkraLong using an extensive collection of long-text question-answering datasets, spanning multiple domains, covering various question types. The experimental results demonstrate that OkraLong not only enhances answer accuracy by 5.7%-41.2%, but also provides superior cost savings of 1.3x-4.7x.

2 Related Work

2.1 Long-Text Processing

Understanding and reasoning over long-form text have always been crucial in natural language processing. Considerable efforts have been made to enhance LLMs to handle long contexts (Tworkowski et al., 2023; Liang et al., 2023; Chen et al., 2023). Besides, increasingly powerful LLMs such as Gemini (Team et al., 2024) and GPT-4 (Achiam et al., 2023), have achieved remarkable large context capability, yet directly processing full-length content incurs high financial expenses.

To address this issue, context compression has emerged as a practical solution for handling large prompts. Extractive compression methods directly select informational tokens or sentences from the context. For instance, RECOMP-Extr (Xu et al., 2024a) performs sentence-level selection based on similarity scores, while LLMlingua (Jiang et al., 2023a; Pan et al., 2024) and Longllmlingua (Jiang et al., 2024) employ token-level filtering through information entropy. Abstractive approaches leverage generative models for content summarization, as exemplified by RECOMP-Abst(Xu et al., 2024a), CompAct (Yoon et al., 2024), and Refiner (Li et al., 2024b). However, these methods still exhibit critical limitations : (1) heavily calling auxiliary models that introduce latency overhead, (2) potential loss of specific information during compression.

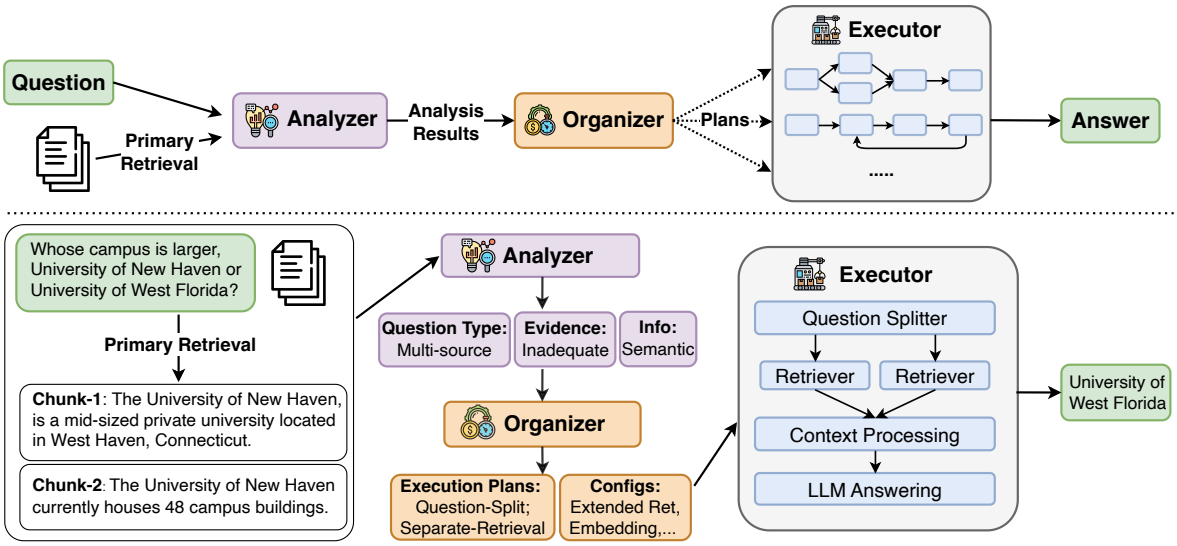

Figure 2: Architecture of OkraLong with an example. After the primary retrieval, the **analyzer** assesses the current task states in three aspects. Based on the analysis, the **organizer** dynamically provides execution plans: the "multi-source" task activates the question-splitter (i.e., prompting the LLM to get two university-specific sub-questions), followed by separate retrieval; then the retrieved contexts should be processed and merged, which are then fed into the LLM for answering. For the configurations, inadequate evidence demands the extended retrieval, while semantic info-pattern favors embedding-based semantic retrieval. Finally, the **executor** carries out the planned workflow. A more detailed example is provided in Appendix B.

2.2 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) is a prevalent technique for enhancing LLM capabilities with external knowledge (Lewis et al., 2020). Conventional RAG pipelines always segment text into chunks, embed them, and retrieve question-relevant content for the LLM generation (Gao et al., 2023). However, basic RAG systems are prone to information loss, particularly in multi-hop queries, leading to suboptimal accuracy (Tang and Yang, 2024; Zhuang et al., 2024; Shao et al., 2023).

Recent advances propose iterative and adaptive refinement mechanisms to mitigate these issues. For instance, FLARE (Jiang et al., 2023b) and DRAGIN (Su et al., 2024) activate the search engine when LLMs output tokens with low probability. Self-RAG (Asai et al., 2024) and MIGRES (Wang et al., 2024) prompt LLMs to make decision on iterative retrieval. Adaptive-RAG (Jeong et al., 2024) and MBA-RAG (Tang et al., 2025) employ adaptive routing strategies to enhance effectiveness. Press et al. (2023) and Gao et al. (2024) improve RAG performance utilizing self-asking and self-correcting. Despite these advancements, in practical settings, existing iterative methods often incur high costs due to extensive LLM calls, and the adaptive strategies remain coarse-grained, failing to account for diverse application scenarios.

3 Methodology

3.1 Framework Overview

In this section, we introduce OkraLong, a flexible and efficient retrieval-augmented framework for long-text question answering. As depicted in Figure 2, OkraLong comprises three core modules: analyzer, organizer and executor. Given a question and the long-form text, OkraLong initiates with primary question-relevant context retrieval. The context and the question are subsequently fed to our lightweight analyzer (implemented as a fine-tuned language model) for real-time task characterization. This analysis covers multiple dimensions, including question types, evidence containing, and information patterns. These analytical results then drive our organizer to dynamically schedule specific execution plans with corresponding configurations. Finally, the executor conducts the optimized processing pipeline through a composition of operators, ultimately generating the final response.

With this architecture, OkraLong facilitates flexible and efficient processing workflows. For flexibility, we develop multiple execution operators as the core infrastructure, and the analyzer provides a comprehensive characterization of task states. These allow for adaptive and fine-grained organization. For efficiency, OkraLong orches-

trates the appropriate processing workflow and budgets, optimizing both response quality and cost-effectiveness. Besides, the lightweight analyzer and the compact framework architecture prevent unaffordable latency, an issue often neglected in previous iterative and generative methodologies.

3.2 Analyzer

The Analyzer constitutes the cognitive foundation of OkraLong, performing real-time assessment of the task states. Its analytical outputs drives dynamic workflow scheduling.

Initially, the analyzer activates the retriever to fetch question-relevant contexts, dispatched alongside the question for assessment. Compared to the pure-question analysis, this inclusive strategy allows for a holistic assessment of both the question requirements and the overall context environment.

The derived analysis results cover three key aspects: (1) Question type (θ_q): Queries with differing objectives demand diverse processing techniques. To organize these, we classify the tasks into five categories: arithmetic, extractive, summarizing, multi-source, and multi-bridge. The multi-source tasks require information from various entities or sources, whereas multi-bridge tasks involve several interconnected procedural steps (more examples in Table 5). (2) Information Pattern (ψ_i): The requisite information can manifest in different forms, such as semantically correlated or exactly matched patterns. We classify these patterns as either semantic, exact, or a combination of both, utilizing it to guide retrieval strategies. (3) Evidence Identification (ϕ_e): We confirm whether the initially retrieved context includes clear evidence to address the question, which also reveals the task's complexity.

Therefore, given long-from text D and the question q, the analyzer can be formulated as:

$$C_q = \{c_1, c_2, ..., c_k\} = \texttt{Retriever}(q, D) \quad (1)$$

$$\{\theta_q, \psi_i, \phi_e\} = \texttt{Analyzer}(\texttt{Instruct}(q, C_q)) \quad (2)$$

where the question q and the retrieved chunks C_q are combined into the instructing prompt. The analyzer returns the θ_q, ψ_i, ϕ_e, which represent the question, information and evidence terms in analytical results.

To implement the analyzer, we refine a lightweight language model via supervised fine-tuning. The training dataset is derived from public datasets. Data entries with human annotations are processed and integrated, while unannotated entries are labeled utilizing the advanced LLM like GPT-4o (more implementation details in Appendix A).

3.3 Organizer

The Organizer serves as the pivotal decision engine that transforms analytical insights into executable plans. It employs a task-aware heuristic orchestration to dynamically optimize and configure the processing pipeline. The organization covers three critical dimensions: workflow architecture, retrieval granularity, and evidence aggregation.

Workflow construction utilizes question-type (θ_q) to organize task-specific processing pipelines, enhancing the targeted handling. For example, multi-bridge questions, which require sequential reasoning across interdependent facts, are managed by a step-wise iterative pipeline that decomposes the task into chained sub-tasks with interleaved retrieval. Multi-source questions trigger a split-aggregate pipeline that independently processes evidence retrieval for distinct entities before final aggregation. Arithmetic questions activate an pipeline with context-extension after the specific retrieval, ensuring both precise detail matching and expansive contextual inclusion. These typology-adaptive workflows guarantee an efficient alignment between query requirement and processing strategy, enhancing both flexibility and efficiency.

Retrieval granularity is adaptively governed through a dual-criteria mechanism. First, the question type (θ_q) naturally dictates the basic retrieval scope and granularity. Contextual tasks (e.g., summarizing questions) would activate extensive context scope, whereas factoid tasks (e.g., extractive questions) and iterative augmented tasks (e.g., multi-bridge questions) adopt more focused narrow context. Additionally, the evidence state (ϕ_e) triggers dynamic granularity adjustments: insufficient evidence initiates a scale extension and granularity expansion to incorporate broader evidential information (more details in Appendix C.3).

Evidence aggregation integrates scores from various retrieval strategies, as no single strategy is universally effective. These strategies primarily include exact sparse retriever (e.g., BM25 (Robertson et al., 2009)) and semantic dense retriever (e.g., various embedding models). Each provides a match score between the question and a candidate text chunk. Guided by the analyzed information pattern (ψ_i), these scores are combined using tailored weights. Specifically, for tasks dominated by ex-

act keywords (e.g., entity lookup), a higher weight w_e is applied to S_{exact}. Conversely, for semantic-centric tasks, the semantic score receives a higher weight w_s. The final aggregated score is computed as: $S = w_e \cdot S_{exact} + w_s \cdot S_{semantic}$. This score, adaptively adjusted to the information pattern, is used to rank and determine the final retrieval results.

Overall, the organizer optimizes both the processing workflow and the modular configurations. These strategic and flexible approaches help to robustly manage complex task scenarios.

3.4 Executor

The executor serves as the core processing engine of OkraLong, comprising multiple distinct operators. Its primary function is to accomplish retrieval-augmented processing, which necessitates basic modules: indexing, retrieval, and generation. To enable more flexible processing pipelines, we enhance these core functionalities and develop the following operators (more details in Appendix C.3):

- Fundamental Operators: Basic text chunking, indexing, context retrieval, and LLM generation.
- Assembled Retriever: This operator integrates multiple retrieval strategies. It normalizes the matching scores and performs weighted aggregation to produce improved context.
- Context Processor: Instead of merely concatenating retrieved text chunks, this operator provides functionalities for context merging, context extension, and table recovery. This ensures both precise detail matching and enriched contextual information.
- Question Splitter: For questions spanning multiple entities, this tool divides the original questions into sub-questions, which are then independently processed and subsequently merged (cf. Table 9).
- Step-wise Reasoner: Complex queries may require step-by-step reasoning. Inspired by iterative processes, it prompts the LLM to perform the current reasoning step and produce next-query for subsequent retrieval operations (cf. Table 10).

These operators, guided by the organizer's execution plans and configurations, empower the executor to support both linear and complex branching processing topologies, thus effectively adapting to diverse task characteristics.

4 Experiment Setups

4.1 Datasets

To evaluate the performance of OkraLong comprehensively, we conduct experiments on six long-text question-answering datasets, spanning various domains and multiple question types:

(1) **FINQA** (Chen et al., 2021) is a financial numerical reasoning dataset, constructed from the earnings reports. (2) **TAT-DQA** (Zhu et al., 2022) is another financial dataset, derived from annual reports, covering diverse question types. (3) **Qasper** (Dasigi et al., 2021) is a reading comprehension dataset based on NLP research papers, containing summarizing and extractive questions. (4) **MultiFieldQA** (Bai et al., 2024) has question-answering pairs sourced from diverse fields, including legal documents, government reports, etc. (5) **HotpotQA** (Yang et al., 2018) involves two-hop questions based on Wikipedia paragraphs. (6) **2WikiMultihopQA** (Ho et al., 2020) consists of multi-hop questions, also based on Wikipedia content.

Detailed characteristics of these datasets are presented in Table 6. To align with practical long-text settings, such as unsegmented full content, we derived these datasets from the processed data collections UDA (Hui et al., 2024) and LongBench (Bai et al., 2024) (further details in Appendix C.1).

4.2 Baselines and Setups

Baselines. We select the following six approaches as the baselines: (1) **Standard RAG** utilizes a traditional chunking, retrieval and generation workflow. (2) **Standard Long-Context Strategy** processes the entire long-text using a LLM without additional context refinement. (3) **LongLLMLingua** (Jiang et al., 2024), a context compression approach that filters tokens based on informational significance according to a lightweight model. (4) **CompAct** (Yoon et al., 2024), another compression approach, employing a lightweight model to iteratively generate the summarized text content. (5) **FLARE** (Jiang et al., 2023b), a dynamic RAG method that adapts retrieval based on token probabilities during iterative text generation. (6) **Adaptive-RAG** (Jeong et al., 2024), another dynamic RAG method, adaptively conducting multi-step or single-step retrieval based on question complexity.

Evaluation Metrics. To assess the quality of the generated responses, we adopt the original evalua-

tion metrics from the source benchmarks (Hui et al., 2024; Bai et al., 2024). For FINQA's numerical-oriented tasks, we employ Exact Match (EM) accuracy, while using F1 scores for all other datasets. Consistent with prior research (Xu et al., 2024a; Li et al., 2024c), we estimate the financial costs by measuring the total token usage of the LLM, as the small retrievers cost negligible in overall evaluation. The latency overhead is recorded as the end-to-end execution time from query submission to final response generation.

Experimental Settings. In all experiments, we utilize the GPT-4o (Hurst et al., 2024) as the backbone LLM for generating answers. Following previous works (Xu et al., 2024a; Asai et al., 2024), we use Contriever-MSMARCO (Izacard et al., 2021) as the basic retrieval model, with a default chunk-size of 512 tokens. For the RAG pipelines, the top-5 chunks are fetched, and for the compression-based pipelines, we fed the top-30 chunks into subsequent compression stages (Yoon et al., 2024).

Implementation Details. We perform supervised fine-tuning on the Llama-3.2-1B-Instruct (Dubey et al., 2024) model to serve as the lightweight analyzer. The fine-tuning dataset was constructed by sampling from the train splits of the HotpotQA, TAT-DQA, and Qasper datasets. This makes the evaluation on the other three datasets out-of-distribution. We employ BM-25 (Robertson et al., 2009), a widely used sparse retriever, for exact retrieval augmentation.

Further details on experiments and implementation are provided in Appendix C.

5 Results and Analysis

5.1 Main Results

Table 1 presents the main experimental results, supported by a visualization of averaged results in Figure 3. Overall, OkraLong demonstrates significant effectiveness in both answering accuracy and cost efficiency. The basic OkraLong maintains decent performance across diverse datasets, although occasional suboptimal results on Multi-FieldQA. This may be attributed to the implicit semantic patterns among its factoid questions, which occasionally challenges the OkraLong's analyzer in task characterization.

Analysis of the baselines reveals two extreme cases: While compression-based CompAct minimizes token usage, its aggressive content summarization causes severe information loss (with 39.9% accuracy degradation). Conversely, leveraging the capability of GPT-4o, long-context processing achieves peak accuracy through exhaustive context retention, but incurs prohibitive costs (16.8x higher than the basic OkraLong).

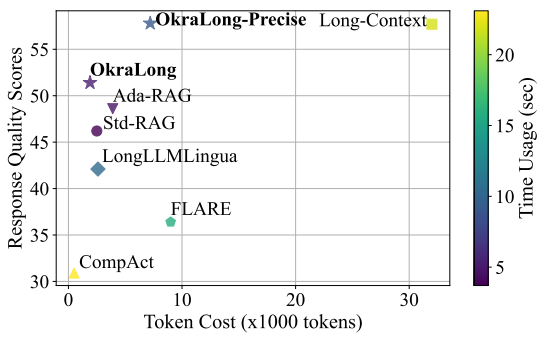

Figure 3: Average performance of end-to-end question answering across six datasets. Superior approaches are **left** and **top** positioned, indicating lower cost and higher accuracy. And the execution time is represented by the colors (the dark color denotes reduced latency).

Aside from these extremes, the basic OkraLong enhances answer accuracy by **5.7%-41.2%** while achieving cost savings of **1.3x-4.7x** compared to prior advancement. Furthermore, we also introduce the OkraLong with **Precise-Mode**, which automatically apply full context to initially unanswerable questions [1]. This cascading augmentation achieves the equivalent answer quality with the long-context processing, while maintaining a **4.4x** cost advantage. The integration of both modes establish a **Pareto-optimal frontier** in the cost-accuracy spectrum (shown in Figure 3), enabling highly efficient deployment in practical long-text query processing.

5.2 Latency Overhead

We conduct a comprehensive latency analysis among different approaches. Figure 4 summarizes the average end-to-end latency results with decomposition.

The overall execution time can be divided into two primary components: context processing and LLM generation. For traditional approaches, the context processing of the standard RAG entails basic indexing and retrieval. The long-context mechanism requires no operations on context, but encounter substantial delays during LLM generation with lengthy input. For compression-based

[1]The generating LLM is prompted to respond "unanswerable" if encountering a lack of evidence.

Method	Average		TAT-DQA		FINQA		Qasper		M-FieldQA		HotpotQA		2WikiMQA	
	Score	**Cost**	F1	Cost	EM	Cost	F1	Cost	F1	Cost	F1	Cost	F1	Cost
Std-RAG	46.2	2.5	43.3	2.9	45.0	2.8	33.8	2.4	55.6	2.2	47.0	2.2	52.3	2.3
LC	**57.8**	32.0	**54.4**	79.8	**56.5**	74.1	**44.9**	10.5	**56.9**	7.2	**63.7**	12.9	**70.1**	7.4
Longllmlingua	42.1	2.6	43.5	3.5	41.4	3.4	34.5	2.5	39.5	1.9	54.0	2.4	39.7	1.9
Compact	30.9	**0.5**	19.0	**0.6**	11.9	**0.6**	19.4	**0.3**	44.3	**0.3**	45.8	**0.5**	45.0	**0.5**
Ada-RAG	48.6	3.9	42.6	5.3	43.9	4.9	36.5	2.7	51.2	2.4	56.3	3.9	60.9	4.2
FLARE	36.4	9.0	31.7	11.2	41.4	10.9	34.8	10.4	45.9	7.4	37.7	6.8	27.0	7.6
OkraLong	51.4	1.9	53.3	1.9	45.3	2.4	36.6	1.8	51.0	1.5	59.5	1.9	62.8	2.2
w/ Precise Mode	**57.8**	7.2	56.7	9.5	56.1	17.8	44.4	4.3	53.5	1.6	63.4	3.1	72.5	2.8

Table 1: End-to-end question answering performance across six datasets. Evaluation scores (F1/EM) are normalized to 0-100 scale for clarity, with the cost quantified as token consumption ($\times 10^3$ tokens) for LLM generation. Performance rankings are indicated with **bold** (for best) and underline (for second best), where the augmented OkraLong with Precise-Mode is independently marked.

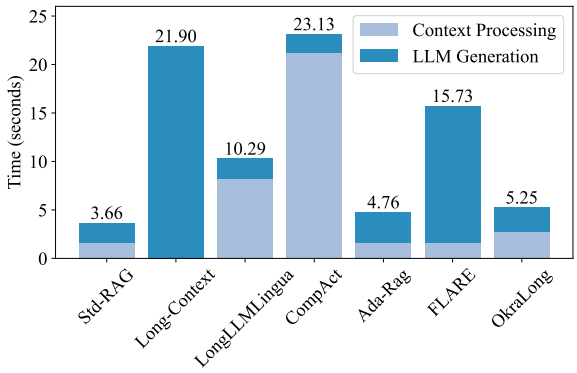

Figure 4: Average end-to-end latency results across various methods. The execution time (per question) comprises context processing time and LLM generation time.

Method	TAT	FIN	Qasper	M-Field	Hotpot	2Wiki
Std-RAG	**7.2**	**7.2**	**2.0**	**1.7**	**2.0**	**1.9**
LC	55.5	57.4	4.7	3.5	6.6	3.7
L-Lingua	16.5	15.8	8.5	6.1	8.5	6.3
CompAct	29.7	32.9	22.3	17.6	20.1	16.3
Ada-RAG	8.9	7.7	3.2	2.4	3.6	2.8
FLARE	31.3	27.7	14.6	6.9	6.0	7.8
OkraLong	8.9	10.5	2.3	2.3	3.6	3.7

Table 2: End-to-end execution latency of different methods across six datasets. Performance rankings are indicated with **bold** (for best) and underline (for second best).

methods, the latency increases as they often rely on small models for iterative compression. Meanwhile, dynamic RAG approaches involve iterative LLM service calls, extending the generation time. Our OkraLong framework adaptively adjusts the workflow with a modest overhead for task analysis and extended indexing. Given the improvements in accuracy and cost efficiency, this marginally extra time is justifiable.

More specific results are shown in Table 2. While standard RAG typically achieves the lowest latency, its rigid processing causes information loss. The financial reports in TAT-DQA and FINQA datasets spanning hundreds of pages, which raises the latency overhead across all methods due to heavy indexing or lengthy full context. For HotpotQA and 2Wikimqa, which require multi-step reasoning, OkraLong spends more time than Qasper and M-FieldQA due to iterative LLM calls. This also reflects the OkraLong's capacity to adapt to diverse demanding.

5.3 Ablation Study

We conduct ablation studies to assess the contribution of various optimizations within OkraLong, using the TAT-DQA dataset for its diverse task characteristics. Table 3 displays the performance changes when removing a specific optimization.

First, disabling adaptive workflow constructions (i.e., using a fixed retrieval-generation pipeline for all tasks) reduces accuracy by 7.3%. While this may reduce token costs, it critically lacks situational adaptability, leading to inefficient processing for diverse tasks. Second, maintaining a fixed moderate retrieval granularity, without dynamically adjusting, results in a 14.1% decrease in accuracy. This significant loss stems from failing to capture critical contextual information. Third, replacing our aggregated retrieval approach with a direct dense retriever causes a 6.4% F1 score drop. This decline is primarily because a single dense retriever

Method	F1 Score	Cost
Std-RAG (baseline)	43.3	2.94
OkraLong	**53.3**	1.93
w/o workflow orchestration	49.4	**1.59**
w/o retrieval adjustment	45.8	1.95
w/o aggregated retrieval	49.9	1.91

Table 3: Ablation studies on the OkraLong framework, assessing the contributions of diverse optimizations.

	Question	Information	Evidence
Analyzer	86.4	64.6	79.4
w/o fine-tuning	22.9	15.2	67.5

Table 4: Prediction accuracy (exact-match scores) across various terms, using fine-tuned analyzer or direct model answering.

struggles to precisely retrieve evidence for specific query details.

These results demonstrate that our synergistic approaches each provide different yet complementary benefits, collectively enhancing OkraLong's overall performance.

5.4 Analyzer Performance

The analyzer forms OkraLong's cognitive core. In this section, we evaluate its performance across three terms: question type classification, information pattern prediction, and evidence identification. Table 4 shows the analyzer's prediction results on the combined validation datasets (more dataset details in A.2).

Compared to directly prompting the small model, supervised fine-tuning significantly improves the prediction performance. Question-type classification achieves a high precision of 86.4%, as a non-trivial five-class categorization task. This efficiency aids in constructing the appropriate workflows, thereby enhancing the overall performance. Evidence identification also performs well with an exact-match score of 79.4%. This facilitates the effective retrieval through dynamic scope adjustment and granularity control. However, the prediction on information pattern shows reduced effectiveness. We attribute this to the inherent complexity to directly predict the optimal retrieval pattern (exact, semantic or both) from the question and contexts, which could exceed the capabilities of a lightweight language model. To mitigate potentially biases, we adopt conservative fusion-weights when integrating the two retrieval strategies (cf. Appendix C.3).

5.5 Robustness and Generalization

OkraLong exhibits robustness and generalization across several dimensions. First, for the patterns, its analyzer is fine-tuned on the heterogeneous dataset (combining HotpotQA, TAT-DQA and Qasper), mitigating over-fitting to specific patterns. This is evidenced by decent end-to-end Q&A performance, across both in-distribution and three **unseen out-of-distribution** datasets. Moreover, our **comprehensive evaluation** covers a wide range: (1) target fields including finance, academia, government reports and general knowledge; (2) long-text forms containing varied single document and concatenated multi-documents; (3) questions ranging from extractive, summarizing, arithmetic, to multi-step reasoning. This extensive range, showcasing its generalization across multiple scenarios.

For enhanced robustness, OkraLong offers a precise mode (Section 5.1) where uncertain or unsolved questions fall back to full-context processing, improving resilience. Additionally, the **modular and plug-in design** further enhances generalization. OkraLong features non-fixed workflow, enabling seamless integration of new operations and heuristics for various real-world requirements. For example, when a small model with limited mathematical capabilities is employed, a code-aided generation operator can be easily incorporated for arithmetic tasks. This ensures OkraLong's extendability for diverse real-world application.

6 Conclusion

In this paper, we propose OkraLong, a flexible and efficient retrieval-augmented framework for long-text question answering. This innovative framework adaptively orchestrates the entire workflow through its three synergistic components: analyzer, organizer, and executor. OkraLong characterizes task states, dynamically organize the workflow, and carries out the execution to generate final answers. We conduct comprehensive evaluations across six diverse datasets, spanning multiple domains and task types. The experimental results indicate that OkraLong not only enhances answering quality but also delivers significant cost-effectiveness. Compared to pre-existing methods, OkraLong demonstrates superior performance in handling long-text questions, thereby providing a highly efficient solution for practical deployment.

Limitations

While OkraLong demonstrates significant improvements in long-text question answering, we acknowledge its limitations: First, to balance accuracy and efficiency, the current analyzer employs supervised fine-tuning of a lightweight model, which relies on annotated training datasets for refinement. Future research could explore semi-supervised or weakly supervised paradigms to further reduce annotation dependence while maintaining effectiveness. Second, while OkraLong efficiently processes textual content, some long-form documents may also require additional multi-modal integration. We currently focus on text-centric workflows, as it remains the primary information carrier. Exploring efficient strategies for querying long-form multi-modal content represents a promising direction for future work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

AzureOpenAI. 2025. Azure openai service.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

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

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, et al. 2021. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*.

Chroma. 2025. Chroma: the ai-native open-source embedding database.

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*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Weizhi Fei, Xueyan Niu, Pingyi Zhou, Lu Hou, Bo Bai, Lei Deng, and Wei Han. 2024. Extending context window of large language models via semantic compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5169–5181, Bangkok, Thailand. Association for Computational Linguistics.

Yuan Gao, Yiheng Zhu, Yuanbin Cao, Yinzhi Zhou, Zhen Wu, Yujie Chen, Shenglan Wu, Haoyuan Hu, and Xinyu Dai. 2024. Dr3: Ask large language models not to give off-topic answers in open domain multi-hop question answering. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5350–5364, Torino, Italia. ELRA and ICCL.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yulong Hui, Yao Lu, and Huanchen Zhang. 2024. UDA: A benchmark suite for retrieval augmented generation in real-world document analysis. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Taeho Hwang, Sukmin Cho, Soyeong Jeong, Hoyun Song, SeungYoon Han, and Jong C Park. 2024. Exit: Context-aware extractive compression for enhancing retrieval-augmented generation. *arXiv preprint arXiv:2412.12559*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong Park. 2024. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. 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 7036–7050, Mexico City, Mexico. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023a. LLMLingua: Compressing prompts for accelerated inference of large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023b. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Xinze Li, Yixin Cao, Yubo Ma, and Aixin Sun. 2024a. Long context vs. rag for llms: An evaluation and revisits. *arXiv preprint arXiv:2501.01880*.

Zhonghao Li, Xuming Hu, Aiwei Liu, Kening Zheng, Sirui Huang, and Hui Xiong. 2024b. Refiner: Restructure retrieval content efficiently to advance question-answering capabilities. *arXiv preprint arXiv:2406.11357*.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024c. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 881–893.

Xinnian Liang, Bing Wang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2023. Unleashing infinite-length input capacity for large-scale language models with self-controlled memory system. *arXiv e-prints*, pages arXiv–2304.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

OpenAI. 2025. Openai api pricing.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274, Singapore. Association for Computational Linguistics.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12991–13013, Bangkok, Thailand. Association for Computational Linguistics.

Xiaqiang Tang, Qiang Gao, Jian Li, Nan Du, Qi Li, and Sihong Xie. 2025. MBA-RAG: a bandit approach for adaptive retrieval-augmented generation through question complexity. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3248–3254, Abu Dhabi, UAE. Association for Computational Linguistics.

Yixuan Tang and Yi Yang. 2024. Multihop-RAG: Benchmarking retrieval-augmented generation for multi-hop queries. In *First Conference on Language Modeling*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2023. Focused transformer: contrastive training for context scaling. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Keheng Wang, Feiyu Duan, Peiguang Li, Sirui Wang, and Xunliang Cai. 2024. Llms know what they need: Leveraging a missing information guided framework to empower retrieval-augmented generation. *arXiv preprint arXiv:2404.14043*.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024a. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2024b. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*.

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, Brussels, Belgium. Association for Computational Linguistics.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. CompAct: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439, Miami, Florida, USA. Association for Computational Linguistics.

Fengbin Zhu, Wenqiang Lei, Fuli Feng, Chao Wang, Haozhou Zhang, and Tat-Seng Chua. 2022. Towards complex document understanding by discrete reasoning. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4857–4866.

Ziyuan Zhuang, Zhiyang Zhang, Sitao Cheng, Fangkai Yang, Jia Liu, Shujian Huang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. 2024. EfficientRAG: Efficient retriever for multi-hop question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3392–3411, Miami, Florida, USA. Association for Computational Linguistics.

A Training Details of the Analyzer

A.1 Question Type

We classify the questions into five categories, encompassing a wide range of practical situations. The detailed descriptions and examples are presented in Table 5.

A.2 Dataset Construction

The training and validation datasets for the analyzer were sampled from the training splits of the HotpotQA, TAT-DQA, and Qasper datasets. Detailed statistics are provided in Table 7.

Each training instance's input consists of the question and relevant document segments, formatted with a specific prompt detailed in Table 8.

Regarding the output terms: (1) Question types, annotated in the original datasets, are standardized into five predefined categories. (2) The information pattern guides retrieval policy selection. First, retrieval is performed independently using exact and semantic methods. Then the context containing more pertinent evidence is designated as the preferred pattern (exact, semantic, or same). (3) Evidence containing is directly labeled according to the annotated evidence and the provided contexts. For datasets without usable evidence annotations (e.g., TAT-DQA), evidence identification was labeled using GPT-4o.

A.3 Training Configuration

We conduct the supervised fine-tuning on Llama-3.2-1B-Instruct model, with LoRA using the following settings:

- Gradient accumulation steps: 64
- Learning rate: 1e-4
- Training epochs: 5
- LoRA rank: 8
- LoRA scaling: 16
- LoRA dropout: 0.1

B Detailed Example of OkraLong Workflow

This section provides a detailed illustration of the OkraLong workflow using a specific example.

- The process begins by inputting the long-text content and the question (e.g., "Which campus is larger, University of New Haven or University of West Florida?"). OkraLong first chunks the context and primarily retrieves relevant chunks related to the question.
- Based on the input question and retrieved chunks, the fine-tuned analyzer outputs the task states, including question type, information pattern, and evidence identification. In this example, the task type is identified as multi-source (requiring information from two separate universities), the information pattern may be semantic-matching, and the initial evidence may be inadequate.
- Based on these states, the organizer will heuristically organize the execution pipeline. The multi-source task activates the question splitter, which prompts the LLM to separate two entities for parallel retrieval. And the semantic-matching pattern favors embedding-based semantic retrieval, while inadequate evidence demands the additional retrieval with extended granularity. After the parallel retrieval, the multi-source informational contexts should be processed and merged. Then the LLM should extract the evidence and generate the final answer.
- Following the above plans, the executor will execute the corresponding operations and output the final answer.

C More Implementation Details

C.1 Experimental Dataset

In our experiments, we utilize the datasets originating from the long-form aligned UDA collection (Hui et al., 2024) and LongBench collection (Bai et al., 2024), adhering to their established configurations.

The UDA collection is released under the CC-BY-SA 4.0 license, while the LongBench collection is covered by the MIT license. Our academic utilization aligns with their designated purposes. These widely-recognized public collections are risk-free without offensive content.

UDA preserves the complete, unsegmented documents along with the source question-answering data points. And LongBench aggregates multiple Wikipedia articles to furnish expansive long-form contexts. The statistics of the test datasets is detailed in Table 6 , illustrating the distribution across two benchmarks.

C.2 Experimental Settings

In our experiments, we employ the GPT-4o model through the AzureOpenAI API (AzureOpenAI,

Question Type	Description	Example Question
Arithmetic	Performing mathematical calculations.	What is the percentage increase in interest expanse and penalties in 2019?
Extractive	Extracting specific information directly from the context.	What crowd-sourcing platform is used?
Summarizing	Involving condensing information from large contexts.	How does the researcher improved the neural network architectures for image recognition?
Multi-Source	Requiring information from various distinct entities or sources.	Which film has the director who was born first, Hell Up In Harlem or The Soviet Story?
Multi-Bridge	Involving a sequence of interconnected procedural steps.	Who is the spouse of the director of film Emergency Wedding?

Table 5: Description of different question types with examples.

Dataset	UDA			Long Bench		
	TAT-DQA	FIN-QA	Qasper	M-FieldQA	HotpotQA	2WikiMQA
Test Size	210	278	232	150	200	200
Data Source	Finance	Finance	Arxiv papers	Multi-field	Wikipedia	Wikipedia
Avg Word Count	72,041	74,170	6,121	4,559	9,157	4,887
Question Types	Arithmetic Extractive Summarizing	Arithmetic	Extractive Summarizing	Extractive	Multi-Souce Mult-Bridge	Multi-Souce Mult-Bridge

Table 6: Detailed characteristics across different datasets. Avg-Word-Count indicates the average number of words in each long-text input.

Dataset	#Train	#Valid
TAT-DQA	3.2k	0.3k
HotpotQA	3.6k	0.4k
Qasper	2.6k	0.3k
Total	9.4k	1.0k

Table 7: Statistics of the training dataset for analyzer. The size of the balanced total dataset is restricted due to limited records in Qasper.

2025), with the API version of 2024-08-06. When evaluate the token cost, we apply a cost-weight to output tokens four times higher than input tokens (OpenAI, 2025). Other small open-source models (serving as the retrieval models or compressing models) are sourced from Huggingface. For our retrieval processes, we utilize ChromaDB (Chroma, 2025) as the vector database. The fine-tuning of our analyzer is conducted on an NVIDIA A100 GPU for an hour, while small model deployments operate on an NVIDIA A10 GPU. The above setup mirrors the general scenario where average individuals deploy lightweight models on limited-capacity GPUs while accessing more powerful LLMs via remote APIs. Following extensive prior works, we conduct experiments with a single run, due to the significant computational cost of LLMs.

When conducting the long-context processing, issues may raise where the complete context surpasses the 128k token limit of the GPT-4o context window. In such cases, we implement a fallback strategy that involves retrieving the top 200 most relevant text chunks with the dense retriever, approximately aggregating to 100k tokens.

C.3 Supplementary Details of OkraLong

Before the analysis, OkraLong primarily retrieves three segments, each comprising 150 tokens, to perform analysis. Subsequent to the analysis, the organizer assigns extended eight text segments to contextual tasks (e.g., summarizing questions), whereas factoid tasks cover five segments. In instances where evidence is analyzed to be absent, the granularity of retrieval-segments scales to 400 tokens for contextual tasks and 256 tokens for factoid tasks.

The execution module of OkraLong comprises multiple operators: (1) In the assembled retriever, we deploy dual retrieval strategies: the semantic dense retriever and the exact sparse retriever. We normalize the relevance scores of the top 20 text chunks using min-max normalization and aggre-

gate them based on semantic or exact information preferences. These preferences influence the final scoring, applying a conservative weight to determine the top-ranked chunks: 3:2 for preferred aggregation or 1:1 for uniform aggregation. (2) The context processor supports context merging, context extension, and table recovery. It maintains essential metadata such as positions and index numbers of text chunks. Utilizing this metadata, it merges neighboring chunks and extends their preceding and succeeding contexts if required. Additionally, it includes a mechanism to detect and recover incomplete tables within the text, leveraging structural markers such as spacing and line breaks. (3) Inspired by previous works (Trivedi et al., 2023; Jiang et al., 2023b; Ma et al., 2023), we prompt the LLM to perform question splitting and step-wise reasoning. The detailed instructions are shown in Table 9 and Table 10.

System:

Given a question and the document context, please answer three questions:

1. What type of question is being asked? The types include: extractive, abstractive, arithmetic, multi-bridge, and multi-source. Extractive means the query is directly factoid; summarizing means the query needs large context and conclusion; arithemtic means the query needs numerical calculation; multi-bridge means the answer requires multiple bridging steps to get the answer;multi-source means the answer requires information from multiple facts (e.g. comparison questions).

2. Is the key information of the question more exact or semantic (according to both the question and the context)? The answer should be "exact", "semantic" or "same".

3. Does the provided context contain the enough information to answer the question? The answer should be either "yes" or "no".

The final answer should be in the format of a dictionary:

{"question-type": "extractive", "info-type": "exact", "containing": "yes"}.

Please strictly follow the format and no explanation is needed.

User:

Context: {context} ### Question: {question} ### Answer:

Table 8: The instructed prompt for the task analyzer.

System:

Given a question, and this question may need the information from multiple sources.

Please split this question into multiple sub-questions, each of which can be answered by a single source. The final answer should be several sub-questions separated by the line-breaker.

Demonstration:

User:

Which university has the larger campus, University of New Haven or University of West Florida?

Assistant:

What is the campus size of University of New Haven?

What is the campus size of University of West Florida?

User:

{question}

Table 9: The instructed prompt for the question splitting operator.

System:

Given a question, which may need multiple steps to get the final answer. Please first get the existing evidence for the question based on the given context, and then generate a next-step query to query additional information. If the question can already be totally answered, you should output '### Answer: The answer is: <answer>' at the end. Otherwise, output 'None'. The answer should be based only on the context. """

Demonstration:

User:

Context: 100 Rifles is directed by Tom Gries and starring Jim Brown and Raquel Welch. ### Question: 100 Rifles is a western film, starring an actress of what nationality?

Assistant:

Evidence: The main actress in 100 Rifles is Raquel Welch. ### Next-Query: What is the nationality of Raquel Welch? ### Answer: None

User:

Context: {context} ### Question:{question}

Table 10: The instructed prompt for the step-wise reasoning operator.