DocSage: An Agentic Information Structuring Workflow for Multi-Document Question Answering

Anonymous ACL submission

Abstract

knowledge-intensive Multi-Document Question Answering (MDQA) requires models to locate, associate, and reason over information scattered across multiple documents while demanding completeness and accuracy. Existing methods, including long-context large language models (LLMs), traditional vector-based retrieval-augmented generation (RAG), and graph-based RAG, struggle to completely and precisely locate scattered key information, often missing critical facts and leading to incorrect answers. To address these issues, we propose DocSage, a schema-guided semantic extraction and dynamic structure adaptation agentic workflow for MDQA. DocSage introduces three core modules: (1) **Active Schema Discovery**, which automatically generates a query-relevant and document-grounded schema that specifies the required entities, attributes, and relations of the query, constraining information retrieval to a highly relevant semantic space; (2) **Structured Information Extraction**, which uses the induced schema as semantic anchors to perform declarative semantic extraction with error guarantees over documents, achieving higher accuracy and interpretability; and (3) **Dynamic Structure Adaptation Reasoning**, which adaptively selects the optimal data structure based on the extracted information and performs reasoning to generate the final answer. DocSage transforms complex MDQA tasks into a structured information processing workflow, precisely localizing question-relevant information and effectively overcoming the coarse-grained retrieval and structural misalignment issues inherent in existing methods. Extensive experiments show that DocSage outperforms all current baselines, achieving substantial improvements in accuracy and evidence recall across diverse tasks. The source code is available at https://anonymous.4open.science/r/DocSage-07A7.

1 Introduction

Multi-Document Question Answering (MDQA) has emerged as a pivotal task in knowledge-intensive natural language processing, underpinning real-world applications such as cross-document financial analysis, legal case retrieval, and scientific literature synthesis (Lin et al., 2025; Wang et al., 2024). Unlike traditional single-document or factoid QA, MDQA requires models to locate, correlate, and reason over information that is often fragmented across multiple unstructured documents. One example question is shown in Figure 1. The inherent complexity of MDQA stems from two challenges: (1) Information fragmentation, where critical facts are scattered across different documents, making it difficult to ensure completeness(Wang et al., 2024); and (2) Structural misalignment, where the disordered and scattered textual form of documents is inconsistent with the structured knowledge representation required for question reasoning(Lin et al., 2025).

Despite significant advances in large language models (LLMs) and retrieval-augmented generation (RAG), existing approaches fall short in addressing these challenges. Long-context LLMs (e.g., GPT-4 Turbo (Achiam et al., 2023), Claude 3 (Anthropic, 2024)) attempt to process entire document collections within a single extended context window. However, empirical studies (Liu et al., 2025a,b) reveal that they suffer from contextual dilution. As input length grows, the model's attention becomes saturated, causing it to miss critical entities or relations, especially when evidence is distributed across many documents. Vector-based RAG (Lewis et al., 2020) mitigates context window constraints by retrieving top-k relevant chunks. Yet its reliance on coarse-grained semantic similarity often retrieves text that is semantically similar but factually unrelated, while omitting structurally critical but lexically distant facts, resulting in incom-

plete evidence chains. More recently, structure-aware RAG methods have been proposed to better organize knowledge. GraphRAG (Edge et al., 2024) constructs a query-agnostic global knowledge graph via LLM-based triple extraction, incurring high computational cost and introducing irrelevant information and noise (Peng et al., 2025). Subsequent community summarization further discards fine-grained details(Zhang et al., 2025), and the static nature of the graph makes it impractical for dynamic document QA(Dong et al., 2026). StructRAG (Li et al., 2024) selects an optimal structure type based on the input question and document summaries, ignoring detailed content. Moreover, its LLM-based extraction and reasoning often fail to capture scattered relevant information as noted in the limitations of LLMs discussed above. These limitations can cause structural misalignment and incomplete evidence capture.

Inspired by the human cognitive principle of moving from abstract to concrete when solving unknown problems, we propose DocSage. As shown in Figure 1, DocSage introduces a three-stage agentic workflow: (1) Active Schema Discovery: generates multiple candidate schemas based on the query and documents, then iteratively refines them using preliminary extraction evidence, ensuring the final schema is both query-relevant and document-grounded. (2) Structured Information Extraction: uses the induced schema as a semantic anchor to extract fact tuples directly from raw documents, with error-aware correction to rectify low-confidence outputs, thereby significantly improving extraction accuracy. (3) Dynamic Structure Reasoning: adaptively selects the optimal reasoning structure based on extracted data characteristics, and performs multi-hop or complex reasoning based on accurate extracted information and deterministic structured operations (e.g., obtaining precise results using SQL) to produce the final answer.

By transforming knowledge-intensive MDQA into a structured information processing pipeline, DocSage has three key advantages: (1) pinpoint fact localization via schema-guided extraction, avoiding the blindness of vector similarity; (2) matching and linking the same entity across different documents through explicit, joinable attributes (e.g., company ID, patient name); and (3) mitigated attention dilution by offloading reasoning to structured operations. We evaluate DocSage on six diverse MDQA benchmarks covering various difficulty levels and document characteristics. Extensive experiments show that DocSage consistently outperforms current methods, achieving significant improvements in both accuracy and evidence recall.

Our main contributions are summarized as follows:

- We propose DocSage, a agentic workflow for MDQA that integrates active schema discovery, error-aware structured extraction, and dynamic structure reasoning, fundamentally addressing the challenges of information fragmentation and structural misalignment.
- We introduce an interactive schema induction mechanism that first generates multiple candidate schemas, then iteratively refines them using preliminary extraction results and active clarification questions, ensuring the final schema is both query-relevant and document-grounded.
- We develop a structured extraction module that uses schema as a semantic anchor for declarative fact extraction directly from raw documents, achieving high accuracy and interpretability while incurring only the storage cost of a lightweight relational database, rather than a vector database.
- We propose a dynamic structure reasoning module that adaptively selects the optimal data structure based on the extracted evidence, and carry out deterministic and reliable multi-step inference.

2 Related Work

2.1 Retrieval-Augmented Generation for MDQA

Standard RAG combines dense passage retrieval with generative models (Lewis et al., 2020). Subsequent works improve retrieval precision via cross-attention (Arora et al., 2023; Izacard and Grave, 2021). However, RAG treats documents as independent units and relies on coarse-grained similarity, often retrieving semantically similar but factually unrelated chunks while missing structurally critical facts, leading to incomplete evidence chains. DOS RAG(Laitenberger et al., 2025) is a simple retrieve-then-read method that preserves original passage order. Its strong performance underscores the importance of document structure. To handle massive indistinguishable documents,

HiQA(Chen et al., 2024) uses cascading metadata and multi-route retrieval. Unlike DocSage, HiQA enhances retrieval through hierarchical context but does not convert documents into a structured, queryable format. IndexRAG(Bao and Shi, 2026) shifts reasoning offline by precomputing bridging facts. This reduces online cost but does not adapt to specific queries; pre-generated facts may not cover all needed connections. DocSage performs query-specific schema induction to discover the minimal schema on the fly.

2.2 Structure-Aware and Graph-Based RAG

GraphRAG(Edge et al., 2024) constructs a query-agnostic global graph via LLM triple extraction, incurring high cost and noise(Peng et al., 2025). Community summarization discards fine details(Chai et al., 2023), and the static graph hinders dynamic updates. GraphRAG can underperform vanilla RAG on some benchmarks(Dong et al., 2026). SLIDERS(Joshi et al., 2026) extracts information into relational databases for SQL-based reasoning, with a data reconciliation stage. However, SLIDERS lacks query-specific schema induction and calibrated confidence estimation. StructRAG(Li et al., 2024) selects a structure type based only on the question and high-level summaries, ignoring detailed content distribution and causing mismatches (e.g., forcing a graph on a task needing a table). Its LLM-based extraction often misses scattered relevant information. DocSage integrates dynamic structure reasoning as an adaptive step after precise Information extraction, aligning the structure with both the query and the extracted evidence.

3 DocSage Framework

DocSage is an end-to-end agentic pipeline that transforms an unstructured document collection $\mathcal{D}$ into a precise, evidence-backed answer for a natural language query q. The framework centers on a three-stage workflow, illustrated in Figure 1.

3.1 Active Schema Discovery

The goal of this module is to derive a minimal, joinable relational schema $\mathcal{S}^q$ that captures the entities, attributes, and relations required to answer q. To achieve this, we use an algorithm **Active Schema Discovery via clarification Queries**(ASD). ASD models schema discovery as an interactive, hypothesis-verification process that generates multiple candidate schemas and iteratively refines them as documents are processed.

Step 1: Initial candidate schema generation. A schema-derivation LLM M_{schema} analyzes q and 3 document samples $\mathcal{D}_{\text{sample}}$ to produce at least K candidate schema hypotheses $\{\mathcal{S}_0^{q,(1)}, \mathcal{S}_0^{q,(2)}, \ldots\}$. Each hypothesis represents a plausible relational decomposition. For example, for a financial comparison query as shown in Figure 1, candidates may include a single table schema (`Company`, `Year`, `Quarter`, `LiabilityItem`, `Amount`, `Unit`), or a temporal embedding schema (`Company`, `Liability2023Q1`, `Liability2023Q2`,.....), etc.

Step 2: Iterative preliminary evidence assessment. The system processes the document collection $\mathcal{D}$ in batches of $B = 5$ documents. For each batch, a lightweight extractor M_{pre} extracts partial tuples from every document under each candidate schema $\mathcal{S}_k^{q,(i)}$; after the batch, each candidate is evaluated on completeness of required attributes extracted and consistency of entities across documents. The detailed calculation method is provided in the Appendix A.1. Based on this evaluation, candidate schemas are revised by merging those with high overlap and similar completeness, pruning those with persistently low completeness or consistency, or splitting a candidate that explains some documents well but fails on others. The revised schema set then proceeds to the next batch, and the process continues until all documents are processed or convergence is reached.

Step 3: Uncertainty-driven clarification question generation. When a candidate schema encounters any of three types of uncertainty signals, the system generates targeted clarification questions P: (1)Entity alignment conflicts: The same entity appears under different names or with inconsistent attribute values. (2) Attribute value anomalies: Extracted values fall outside reasonable ranges or show logical contradictions.(3) Missing relationships: The current schema cannot establish the multi-hop connections required by the query. The question generator(LLM) formulates questions based on the most plausible candidates.

Step 4: Schema update based on retrieved answers. The question set P is used as new "queries" to retrieve evidence from the related documents. Retrieved answers feed the update function

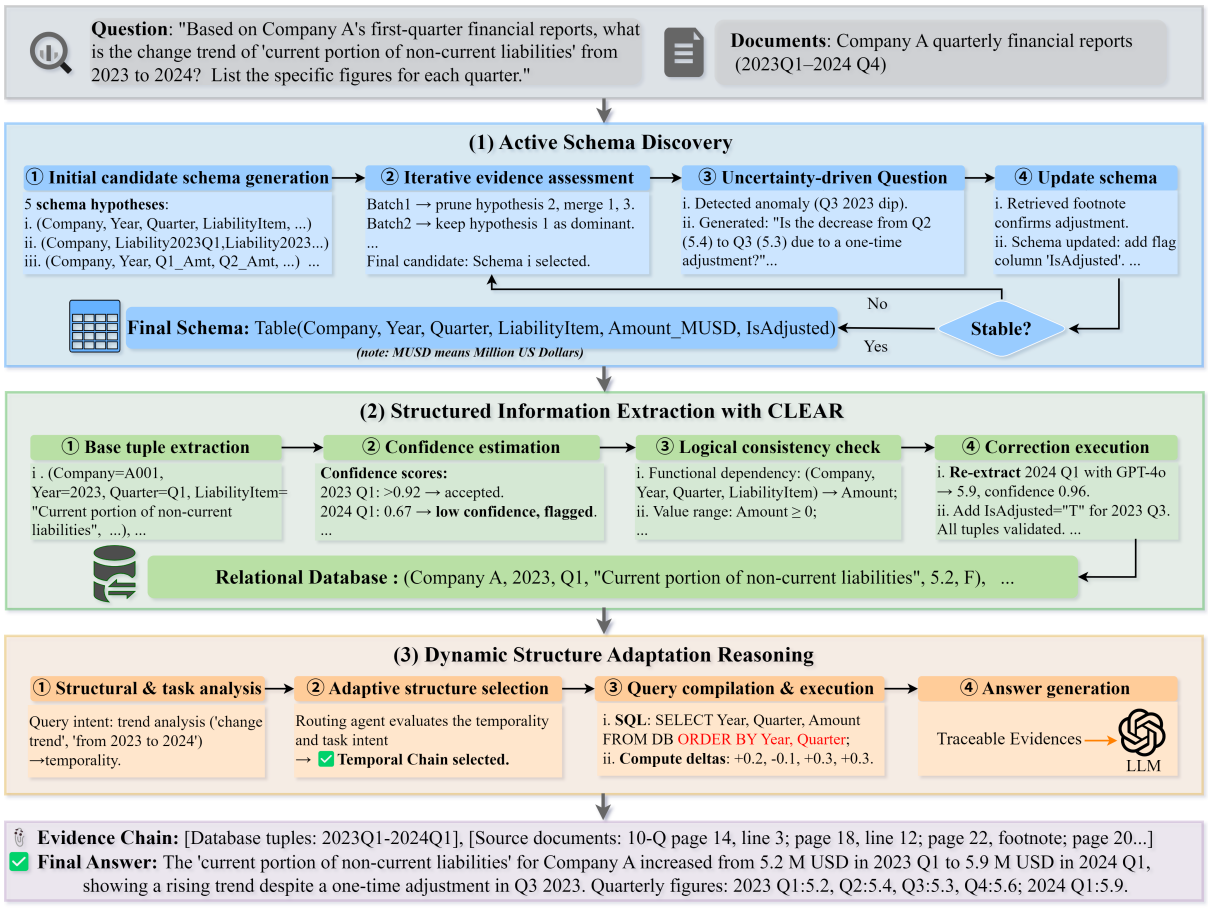

Figure 1: The overview of DocSage pipeline. DocSage transforms a document collection and a query into an answer through three stages: (1) Active Schema Discovery iteratively induces a query-relevant and document-grounded schema; (2) Structured Extraction extracts and corrects tuples with statistical confidence and logical constraints; (3) Dynamic Structure Reasoning selects a temporal chain based on data characteristics in this case: discrete (timestamp, value) pairs are organized into an ordered sequence, and deterministic operations (compute deltas) are applied here. Answer is then generated with traceable evidences.

$\mathcal{F}_{\text{update}}$ to revise each uncertain schema:

$$\mathcal{S}_{k+1}^{q,(i)} = \mathcal{F}_{\text{update}}\big(\mathcal{S}_{k}^{q,(i)},\ \mathcal{A}(P, \mathcal{D}_{related})\big),$$

where $\mathcal{A}$ denotes the answer retrieval process.

Step 5: Convergence and output. When the candidate set stabilizes, which means no additions, merges, or prunings occur across two consecutive batches, and all generated questions have been sufficiently answered, the system outputs the final schema $\mathcal{S}^q$. This schema is both query-relevant and document-grounded.

3.2 Structured Information Extraction

This module employs a **Cross-record Logic Enforcement for Accuracy Reinforcement** (CLEAR) algorithm integrated with confidence estimation to directly extract high-fidelity tuples with provable error guarantees from raw documents, thereby populating the target schema $\mathcal{S}^q$. The core idea of CLEAR is to combine statistical confidence estimation with cross-record logical constraints to achieve dual quality control over extraction results. It first uses LLM hidden states to train lightweight classifiers, producing a calibrated confidence score for each extracted tuple. At the same time, it automatically derives logical rules like functional dependencies, value ranges, temporal orders from the schema and validates the consistency of all tuples. Tuples with low confidence or that violate constraints trigger correction and re-extraction. The key insight is that pointwise confidence is susceptible to noise, whereas cross-record logic provides an independent correction signal; together they significantly improve reliability.

Step 1: Base tuple extraction. For each document $d \in \mathcal{D}$, DocSage extracts candidate tuples

simultaneously for all candidate schemas. Specifically, the document d is concatenated with the descriptions of every candidate schema into a single prompt. This prompt is fed once to a small extraction model M_{ext}, which is instructed to a tuple conforming to that schema. Parallel extraction compresses the entire operation into a single LM call, reducing inference time and token costs.

Step 2: Confidence estimation: hidden states conformal prediction. To quantify the correctness of each extracted value, inspired by RDD's SCAPE-Hyb (Chao et al., 2025), we adopt the following method:

(1) **Hidden state capture**: While M_{ext} generates an tuple, we record hidden states at all Transformer layers. For the output token sequence $\mathbf{w} = (w_1, \ldots, w_m)$ and each selected layer $l \in \mathcal{L}$ (typically the last 4–6 layers), compute the mean and max of token-level hidden states and concatenate them into a fixed-length vector:

$$h_{k,i}^{(l)} = \text{concat}\big(\text{mean}_t h^{(l)}(w_t),\ \max_t h^{(l)}(w_t)\big) \in \mathbb{R}^{2d},$$

where d is the hidden dimension. Vectors from all layers are concatenated to form the feature representation.

(2) **Layer-wise classifiers**: Build a small labeled dataset $\mathcal{D}_{\text{cls}}$ (50–150 examples) where each example is an extracted value with a correctness label (correct/incorrect). For each layer l, train a lightweight logistic regression classifier $f^{(l)} : \mathbb{R}^{2d} \to [0, 1]$ that outputs the error probability $\pi^{(l)} = f^{(l)}(h^{(l)})$.

(3) **Conformal prediction with disagreement signal**. The steps are as followed:

- Split $\mathcal{D}_{\text{cls}}$ into a calibration set $\mathcal{D}_{\text{cal}}$.
- For each calibration sample, compute non-conformity vectors $\mathbf{s}(c) \in \mathbb{R}^{|\mathcal{L}|}$, where $s_l(1) = 1 - \pi^{(l)}$ (assuming error) and $s_l(0) = \pi^{(l)}$ (assuming correct).
- Cluster all $\mathbf{s}(1)$ vectors via k-means to obtain cells $C_1, \ldots, C_K$. Choose the smallest set of top-ranked cells such that the proportion of true errors covered is at least $1 - \alpha$ (user-specified coverage).
- Compute a disagreement signal $\Delta = \max_l |\pi^{(l)} - \bar{\pi}|$ where $\bar{\pi}$ is the average probability across layers. Augment the non-conformity vector to $\mathbf{s}_\lambda(c) = [(1 - \lambda)s_1(c), \ldots, (1 - \lambda)s_{|\mathcal{L}|}(c), \lambda\Delta]$ with $\lambda \in [0, 1]$ (default 0.5). Repeat clustering and selection on the augmented space.
- At test time, for a new extraction, compute $\mathbf{s}_\lambda(1)$ and $\mathbf{s}_\lambda(0)$. If $\mathbf{s}_\lambda(1)$ falls into the selected cells, the extraction is flagged as erroneous and requires correction; otherwise it is accepted.

Step 3: Cross-record logical consistency checking. Once the final schema $\mathcal{S}^q$ is determined, the constraint-derivation LLM $M_{\text{constraint}}$ automatically generates a set of logical constraints (e.g., functional dependencies, value ranges, temporal ordering, referential integrity) from the schema definition and the query. During extraction, all candidate tuples are validated against these constraints; any tuple that violates a constraint is immediately flagged for correction, ensuring that the extracted information adheres to both schema semantics and cross-record consistency.

Step 4: Correction execution. For any tuple flagged as low-confidence by the conformal prediction step or identified as violating an automatically generated constraint, the system triggers a targeted correction workflow. Simple low-confidence or range violations are resolved by re-extracting the same document chunk with a more powerful LLM (e.g., GPT-4o). Complex conflicts such as functional dependency violations or referential integrity mismatches are also handled by the LLM with a verification and disambiguation prompt(Appendix A.2) that performs deeper contextual analysis or targeted backtracking retrieval. Unit inconsistencies are automatically normalized. After correction, all verified tuples are inserted into the high-quality relational database $\mathcal{DB}^q$.

3.3 Dynamic Structure Reasoning

Execute the query q on the constructed database $\mathcal{DB}^q$ by adaptively choosing the optimal data structure and performing inference.

Step 1: Adaptive structure selection. he system employs a LLM as a routing agent. The agent analyzes summary of $\mathcal{DB}^q$ and the implicit requirements of the original query q along the following dimensions: relation density, attribute comparability, temporality, and task intent (e.g., statistical comparison, multi-hop reasoning, planning, hierarchical summarization, or simple fact lookup).

Based on the analysis, the routing agent selects the optimal structure from a candidate set that includes relational tables (for statistical aggregation), knowledge subgraphs (for multi-hop inference), temporal chains (for trend analysis), algorithms (for planning), catalogues (for hierarchical summarization), and plain text chunks (for simple fact lookup).

Step 2: Query compilation and execution. A reasoning LLM M_{reason} compiles the natural language query into deterministic operations on the chosen structure. For a relational table, this yields an SQL query; for a knowledge subgraph, a graph traversal statement (e.g., Cypher); for a temporal chain, a sequence of operations (e.g., sort, sliding window). The optimized plan is denoted as:

$$Q_{\text{opt}} = M_{\text{reason}}(q, \mathcal{S}^q, \text{structure_type}).$$

Step 3: Answer generation and traceability. Executing Q_{opt} yields a structured result set R. The system automatically traces each element of R back to its source tuples in $\mathcal{DB}^q$ and further to the original document location $\mathcal{D}_{\text{context}}$. The reasoning LLM $\mathcal{F}_{\text{synthesize}}$ produces the final answer a with the complete evidence chain $\mathcal{E}$:

$$a,\ \mathcal{E} = M_{\text{reason}}\big(R,\ \mathcal{DB}^q,\ \mathcal{D}_{\text{context}}\big).$$

This design ensures that every claim in the answer is verifiable, eliminates attention diffusion, and reduces hallucinations of M_{reason}.

4 Experiment

4.1 Experimental Setup

Evaluation Datasets. We evaluate on six challenging MDQA benchmarks. (1) **MEBench** (Lin et al., 2025) contains 4,780 questions testing multi-entity reasoning. (2) **Loong** (Wang et al., 2024) includes four tasks with document lengths from 100K to 200K tokens, assessing information localization as documents lengthen. (3) **MuDABench** (Li et al., 2026) comprises over 80,000 pages and 332 analytical instances requiring cross-document aggregation. (4) **ASTRA-QA** (Wang et al., 2026) offers 869 abstract QA instances with faithfulness annotations. (5) **LongBench v2** (Bai et al., 2025) provides 503 challenging multiple-choice questions up to 2M tokens. (6) **DocHop-QA** (Park et al., 2025) is a 11,379-instance multimodal, multi-hop benchmark over scientific documents.

Implementation Details. DocSage uses GPT-4o as primary LLMs, Mistral-7B (Jiang et al., 2023) as the small-scale extraction model, and GPT-4o as the reasoning model.

Baselines. We compare against representative methods across different technical routes. For long-context and standard retrieval methods, we include **GPT-4o** (Achiam et al., 2023), **standard RAG** (Lewis et al., 2020), and **DOS RAG** (Laitenberger et al., 2025). For graph-based and structure-aware methods, we include **GraphRAG** (Edge et al., 2024) and **StructRAG** (Li et al., 2024). For multi-document QA system, we include **HiQA** (Chen et al., 2024) which uses cascading metadata and multi-route retrieval. For structured reasoning, we include **SLIDERS** (Joshi et al., 2026). For offline indexing, we include **IndexRAG** (Bao and Shi, 2026). This selection spans LLMs, standard RAG, graph RAG and structure-aware methods, multi-document frameworks, structured reasoning, and offline indexing, enabling a comprehensive evaluation.

Evaluation Metrics. MEBench uses overall accuracy; Loong uses LLM-judged score (0–100) and exact match (EM) rate; MuDABench uses final answer accuracy and intermediate fact coverage; ASTRA-QA uses topic coverage and unsupported content score; LongBench v2 uses multiple-choice accuracy; DocHop-QA uses generative F1 and index prediction accuracy. The metrics are listed in Table 1.

4.2 Main Results

Based on the comprehensive experimental results reported in Table 1, we draw the following three main conclusions:

1) **DocSage fundamentally resolves the structural misalignment problem in MDQA, delivering especially large gains in multi-entity reasoning and cross-document aggregation.** On MEBench, DocSage achieves 89.2% accuracy, outperforming StructRAG (61.2%) and standard RAG (62.0%) by over 27 percentage points. On MuDABench, which requires analytical aggregation across >80,000 pages, DocSage attains 67.3% final answer accuracy and 78.2% intermediate fact coverage, improvements of +13.5% and +15.8% over StructRAG, respectively. These results confirm that schema-guided extraction, as opposed to coarse-grained similarity retrieval or static graph construction, is critical for precisely localizing and linking fragmented information.

Table 1: Overall main results across all benchmarks. DocSage consistently outperforms all baselines on multi-document QA tasks of varying complexity, document scale, and modality.

Dataset	Metric	GPT-4o	RAG	GraphRAG	StructRAG	HiQA	SLIDERS	IndexRAG	DocSage
MEBench	Acc (%)	33.8	62.0	58.6	61.2	60.4	59.7	58.1	**89.2**
Loong	Avg (0-100)	60.11	46.11	40.82	69.43	67.25	65.87	63.92	**68.29**
	EM (%)	29.2	23.3	18.6	35.1	32.1	28.5	26.8	**53.3**
MuDABench	Final Acc (%)	–	28.4	–	53.8	49.7	51.3	47.6	**67.3**
	Fact Cov (%)	–	41.2	–	62.4	59.1	60.8	55.3	**78.2**
ASTRA-QA	Topic Cov (%)	58.3	62.1	–	68.4	66.8	65.2	63.7	**82.6**
	Unsup (%)	72.6	75.4	–	79.2	77.5	76.1	74.8	**91.3**
LongBench v2	Acc (%)	42.7	38.2	34.5	48.6	46.8	45.3	44.1	**54.3**
DocHop-QA	F1 (0-1)	0.54	0.498	0.445	0.603	0.587	0.571	0.562	**0.687**
	Index Acc (%)	68.7	63.4	58.2	72.5	70.8	69.6	68.9	**79.1**

Note: "–" indicates that the method was not evaluated or not applicable.

Table 2: Average token consumption and runtime of all methods on Loong Benchmark

Method	Average Tokens (k)	Average Runtime
GPT-4o	200.8	16.2
RAG	3.3	5.5
GraphRAG	16.2	25.0
StructRAG	13.5	8.6
HiQA	6.9	9.0
SLIDERS	7.0	11.0
IndexRAG	3.7	4.2
DocSage	**7.5**	**22**

2) **The advantage of DocSage scales with document length and information dispersion.** On Loong, where documents range from 100K to 250K tokens, DocSage achieves the highest exact match rate (53.3%). Notably, on the longest split (200K–250K tokens), its EM rate (47%) far exceeds all baselines (the next best, StructRAG, achieves 10%). This demonstrates that dynamic structure reasoning, combined with error-aware extraction, effectively mitigates attention dilution and evidence scattering, two critical challenges that severely degrade long-context LLMs and static structure-aware methods.

3) **Faithfulness and retrieval precision are substantially improved through CLEAR's dual quality control.** On ASTRA-QA, DocSage achieves the highest topic coverage (82.6%) and the lowest unsupported content ratio (91.3%, i.e., fewest hallucinations), surpassing StructRAG and GPT-4o. On LongBench v2 (up to 2M tokens), DocSage reaches 54.3% accuracy, approaching the human expert performance (53.7%) and exceeding all baselines by 5.7–16.1 points. These results, together with the extraction error rate reduction from 12.4% to 0.7% (Figure 2), establish that active schema discovery and confidence-calibrated logical constraints are essential for producing verifiable, hallucination-resistant answers in ultra-long and knowledge-intensive scenarios.

4.3 Efficiency Analysis

We evaluate the online token consumption and runtime of each method, including the average total number of LLM tokens (input + output) and average time required from receiving a query to get an answer, with all LLM calls made using the OpenAI API. Offline preprocessing (e.g., index building for IndexRAG, graph precomputation for GraphRAG) is not counted. All measurements were conducted on 8 NVIDIA RTX 4090 GPUs (each with 24 GB VRAM). The task uses Loong dataset.

From Table 2, we draw the following two main conclusions: (1) **DocSage achieves an accuracy–efficiency trade-off: it attains the highest accuracy among all methods within a manageable latency.** For example, compared with StructRAG (13.5k tokens, 8.6 s), DocSage improves MEBench accuracy from 61.2% to 89.2% at the cost of approximately 2.6× higher latency and 0.63× more token consumption, demonstrating that its efficiency overhead is fully compensated by the accuracy gain. (2) **DocSage's efficiency bottleneck is full-document scanning with a small model, not expensive LLM calls, leaving clear room for further optimization.** Of the total 22s latency, about 18s come from scans of all docu-

Table 3: Results of the ablation study on the MEBench benchmark.

Method	Overall
DocSage (Full)	**0.892**
w/o Structured Extraction	0.691
w/o Dynamic Structure Reasoning	0.773
w/o Schema Discovery	0.781
w/o CLEAR (Logic Checking)	0.849

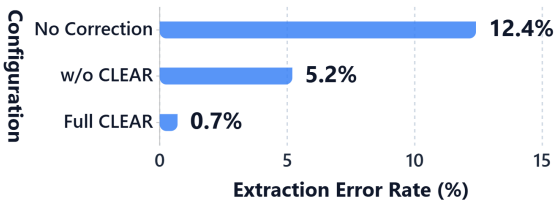

Figure 2: Full CLEAR achieves error rate 0.7% .

ments using Mistral-7B, and only about 4-5s from on-demand GPT-4o.

4.4 Ablation Study

To validate the effectiveness of each core component in DocSage, we conduct systematic ablation experiments on the MEBench dataset. Results are shown in Table 3.

Removing the Structured Extraction module causes the largest performance drop (from 0.892 to 0.691, a decrease of 20.1 percentage points), confirming it as the cornerstone of the framework. Removing Dynamic Structure Reasoning reduces accuracy by 11.9%, and removing Schema Discovery by 11.1 points. Disabling the logical consistency check in CLEAR (w/o CLEAR) lowers accuracy by 4.3%. These results indicate that structured extraction contributes the most, followed by schema-guided reasoning and schema discovery; both logical checking and active questioning further improve robustness. Additional ablation results on the Loong benchmark are provided in Appendix A, which consistently confirm the effectiveness of each module.

4.5 Extraction Error Rate Analysis.

To directly quantify the improvement brought by the CLEAR mechanism, we randomly sample 300 tuples requiring extraction from the Statistics subset of MEBench and manually label their correctness. Three configurations are compared: No Correction (base extraction model only), w/o CLEAR (confidence estimation with conformal prediction but without cross-record logical constraints), and Full CLEAR. Results are presented in Figure 2. The results validates that the dual control of CLEAR, statistical confidence plus logical consistency, significantly reduces extraction errors, thereby supporting DocSage's high-precision answer generation.

5 Conclusion

We have presented DocSage, a framework for multi-document question answering that addresses the core challenges of information fragmentation and structural misalignment. DocSage introduces three synergistic modules. Active Schema Discovery generates and iteratively refines multiple candidate schemas using both the query and document content, actively resolving ambiguities through targeted clarification questions. Structured Extraction with CLEAR performs declarative extraction with confidence estimation and logical constraints, reducing extraction errors below 1%. Dynamic Structure Reasoning adaptively selects the optimal reasoning structure and performs deterministic multi-step inference, producing verifiable answers with full provenance. Extensive experiments on six diverse benchmarks demonstrate that DocSage consistently outperforms a wide range of baselines. DocSage achieves average absolute improvements in accuracy and evidence recall across tasks, while maintaining acceptable online token consumption and runtime. Ablation studies confirm that each module contributes substantially to the overall performance, with structured extraction being the most critical component.

6 Limitations

DocSage's current design incurs non-trivial online latency due to scanning all documents with a small language model. Future work can explore document pre-filtering, incremental schema maintenance, or hybrid offline-online strategies to reduce runtime. The framework also assumes a degree of semantic coherence in source documents; extremely noisy or contradictory corpora may challenge extraction accuracy. Further directions include extending DocSage to multimodal documents, integrating more advanced error-aware techniques, and optimizing monetary cost for real-world deployment at scale.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2024. Claude 3 api documentation. Online. Supports Opus, Sonnet, and Haiku models :cite[5]:cite[8].

Daman Arora, Anush Kini, Sayak Ray Chowdhury, Nagarajan Natarajan, Gaurav Sinha, and Amit Sharma. 2023. Gar-meets-rag paradigm for zero-shot information retrieval. *arXiv preprint arXiv:2310.20158*.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3639–3664, Vienna, Austria. Association for Computational Linguistics.

Zhenghua Bao and Yi Shi. 2026. Indexrag: Bridging facts for cross-document reasoning at index time. *Preprint*, arXiv:2603.16415.

Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. 2023. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint arXiv:2310.05845*.

Daren Chao, Kaiwen Chen, Naiqing Guan, and Nick Koudas. 2025. Relational deep dive: Error-aware queries over unstructured data. *Preprint*, arXiv:2511.02711.

Xinyue Chen, Pengyu Gao, Jiangjiang Song, and Xiaoyang Tan. 2024. Hiqa: A hierarchical contextual augmentation rag for multi-documents qa. *Preprint*, arXiv:2402.01767.

Su Dong, Qinggang Zhang, Yilin Xiao, Shengyuan Chen, Chuang Zhou, and Xiao Huang. 2026. Use graph when it needs: Efficiently and adaptively integrating retrieval-augmented generation with graphs. *arXiv preprint arXiv:2602.03578*.

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

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Harshit Joshi, Priyank Shethia, Jadelynn Dao, and Monica S. Lam. 2026. Contexts are never long enough: Structured reasoning for scalable question answering over long document sets. *Preprint*, arXiv:2604.22294.

Alex Laitenberger, Christopher D Manning, and Nelson F. Liu. 2025. Stronger baselines for retrieval-augmented generation with long-context language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 32559–32569, Suzhou, China. 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, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Zhanli Li, Yixuan Cao, Lvzhou Luo, and Ping Luo. 2026. Navigating large-scale document collections: Mudabench for multi-document analytical qa. *Preprint*, arXiv:2604.22239.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. 2024. Structrag: Boosting knowledge intensive reasoning of llms via inference-time hybrid information structurization. *Preprint*, arXiv:2410.08815.

Teng Lin, Yuyu Luo, Honglin Zhang, Jicheng Zhang, Chunlin Liu, Kaishun Wu, and Nan Tang. 2025. MEBench: Benchmarking large language models for cross-document multi-entity question answering. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1481–1494, Suzhou, China. Association for Computational Linguistics.

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

Yijun Liu, Jinzheng Yu, Yang Xu, Zhongyang Li, and Qingfu Zhu. 2025b. A survey on transformer context extension: Approaches and evaluation. *CoRR*, abs/2503.13299.

Jiwon Park, Seohyun Pyeon, Jinwoo Kim, Rina Carines Cabal, Yihao Ding, and Soyeon Caren Han. 2025. Dochop-qa: Towards multi-hop reasoning over multimodal document collections. *Preprint*, arXiv:2508.15851.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2025. Graph retrieval-augmented generation: A survey. *ACM Transactions on Information Systems*, 44(2):1–52.

Minzheng Wang, Longze Chen, Fu Cheng, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, Yunshui Li, Min Yang, Fei Huang, and Yongbin Li. 2024. Leave no document behind: Benchmarking long-context LLMs with extended multi-doc QA. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5627–5646, Miami, Florida, USA. Association for Computational Linguistics.

Shu Wang, Shansong Zhou, Xinyang Wang, Shiwei Wang, Hulong Wu, and Yixiang Fang. 2026. Astra-qa: A benchmark for abstract question answering over documents. *arXiv*.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Hao Chen, Yilin Xiao, Chuang Zhou, Junnan Dong, and 1 others. 2025. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*.

A Appendix

A.1 Completeness and Consistency Evaluation.

After processing a batch of documents (e.g., 5 documents), the system evaluates each candidate schema $\mathcal{S}_k^{q,(i)}$ on two metrics: **completeness** and **consistency**.

Completeness. Completeness measures the proportion of query-required attributes that have been successfully extracted so far. Let A_{required} be the set of attributes required by the query (e.g., company name, period, liability item, amount, unit). The system maintains an attribute occurrence counter for each candidate schema; whenever a non-null value for an attribute is extracted from a document chunk, that attribute's counter is incremented (multiple extractions of the same attribute count only once). Completeness is defined as:

$$\text{completeness}(\mathcal{S}) = \frac{|\{a \in A_{\text{required}} \mid \text{count}(a) > 0\}|}{|A_{\text{required}}|}.$$

If completeness falls below 0.6 for two consecutive batches, the candidate is pruned; candidates that reach completeness 1.0 are considered answerable.

Consistency. Consistency evaluates whether extracted values for the same entity across different documents are compatible. It consists of two sub-dimensions: intra-entity consistency and inter-entity (referential) consistency.

Intra-entity consistency. For each entity e and each attribute a, the system collects all extracted values $\{v_1, v_2, \ldots, v_n\}$ from different documents. Values are first normalized (e.g., amounts converted to a common numeric unit, dates standardized, terminology mapped to canonical names using a knowledge base obtained from prior clarification questions).

- For numerical attributes, the relative standard deviation is computed: $\text{RSD} = \sigma/\mu$. If $\text{RSD} < 0.05$, the values are considered consistent; otherwise a conflict is flagged.
- For categorical attributes, normalized strings are compared. If all values are identical, or if they differ but have been confirmed as synonyms by previous clarifications, they are consistent; otherwise a conflict is flagged.

The intra-entity consistency score for entity e and attribute a is:

$$\text{score}_{\text{ent}}(e, a) = \begin{cases} 1 & \text{all values are consistent,} \\ 0 & \text{an unresolved conflict exists.} \end{cases}$$

Inter-entity consistency. For schemas with foreign-key relationships (e.g., FinancialData.CompanyID must exist in the Company table), the system checks referential integrity. Let ID_{fact} be the set of extracted foreign keys and ID_{dim} the set of primary keys from the referenced table. The missing ratio is:

$$\text{missing}_{\text{fk}} = \frac{|\text{ID}_{\text{fact}} \setminus \text{ID}_{\text{dim}}|}{|\text{ID}_{\text{fact}}|}.$$

If $\text{missing}_{\text{fk}} > 0.1$, referential consistency is considered poor.

Overall consistency. The overall consistency of a candidate schema is the minimum of all intra-entity scores and the referential integrity component:

$$\text{consistency}(\mathcal{S}) = \min\Big(\min_{e,a} \text{score}_{\text{ent}}(e, a),\ 1 - \text{missing}_{\text{fk}}\Big).$$

A consistency score below 0.7 triggers uncertainty signals, leading to the generation of clarification questions.

Using the evaluation results. The system maintains a sliding window (last three batches) of completeness and consistency scores for each candidate schema. Revision decisions are made as follows:

- If completeness < 0.6 and consistency < 0.5 for two consecutive batches, the candidate is pruned.
- If two candidates have very similar completeness and consistency scores and their extracted tuples heavily overlap, they are merged (keeping the simpler table structure).
- If a candidate achieves high completeness on some documents but very low on others, it is split into two variants (e.g., based on document subsets).

When consistency drops due to entity conflicts or missing foreign keys, targeted clarification questions (e.g., "Is the amount difference between Document A and Document B due to different units?") are generated and added to the question queue.

You are a verification and disambiguation assistant for a multi-document extraction system. You are given a conflict involving one or more extracted tuples, the corresponding source document chunks, the target schema, and the conflicting constraint.
Your task:
1. Analyze the provided evidence and determine the most likely correct value(s) based on the source text.
2. If the evidence is insufficient, propose a backtracking retrieval query to locate additional relevant document chunks that could resolve the conflict.
3. Output a structured decision: either a corrected tuple (or set of tuples), or a specific retrieval request.
Conflict description: {conflict_type}
(e.g., "Functional dependency violation: Same CompanyID has two different Amount values for same Period and LiabilityItem", "Referential integrity mismatch: Foreign key 'CompanyID' value X not found in Company table")
Conflicting tuples (with provenance): {conflicting_tuples_with_source}
Target schema: {schema_description}
Conflicting constraint: {constraint_definition}
Source text snippets (from the documents where the tuples originated): {source_text}
Instructions:
- If one source clearly outweighs the other (e.g., contains explicit numeric amounts with clear unit, the other appears to be a misinterpretation), output the corrected tuple as:
CORRECTED_TUPLE: (tuple)
- If both are plausible but contradictory, and additional context could resolve the conflict, output a retrieval request as:
RETRIEVAL_NEEDED: (a short natural language query to find decisive evidence from other parts of the document collection)
- If neither is correct, output: NO_CORRECTION_AVAILABLE: (reasoning)
Output format, strictly follow one of the three patterns:
CORRECTED_TUPLE: (attribute1=value1, attribute2=value2, ...)
RETRIEVAL_NEEDED: "exact query phrase"
NO_CORRECTION_AVAILABLE: "explanation"

Figure 3: The prompt for the Verification and Disambiguation module.

A.2 The prompt for the Verification & Disambiguation module

The prompt is shown in Figure 3.

A.3 Ablation Study Results

A.3.1 Ablation Study Results for DocSage on Loong

Analysis of Results The ablation results on the Loong benchmark, which specifically tests information localization and reasoning in long documents, further validate the efficacy of DocSage's core design. The full model consistently outperforms all ablated variants across tasks and document length categories, with the performance gap widening notably in the most challenging, longest document setting (200K-250K tokens).

1. **Schema Discovery is Critical for Long Documents**: Removing the Schema Discovery module (**w/o Schema Discovery**) causes the most severe performance degradation, especially in **Chain of Reason** (-13.22 points overall) and **Clustering** tasks. This underscores that discovering a coherent relational structure is paramount for organizing and connecting information scattered across hundreds of thousands of tokens.

2. **Structured Extraction Enables Robust Comparisons**: The variant **w/o Structured Extraction** suffers significantly in **Comparison** and **Clustering** tasks (-9.17 and -9.20

Table 4: Ablation Study Results for DocSage on Loong Benchmark

Method	Spotlight Locating	Comparison	Clustering	Chain of Reason	Overall
DocSage (Full)	**85.06**	**73.28**	**64.43**	**73.67**	**68.29**
w/o Schema Discovery	73.14	58.92	51.87	60.45	59.10
w/o Structured Extraction	78.35	64.11	55.23	66.88	63.64
w/o Schema-Guided Reasoning	80.02	67.45	58.91	68.52	65.73
w/o CLEAR (Logic Checking)	82.57	70.33	61.05	70.89	66.71
w/ Passive Schema Discovery	83.41	71.16	62.78	72.05	67.35

points overall), which require precise entity matching and attribute aggregation. This confirms that converting text into clean, query-aligned tuples is more reliable than reasoning over raw, noisy text chunks.

3. **Schema-Guided Reasoning Mitigates Context Overload**: While the drop for **w/o Schema-Guided Reasoning** is slightly smaller on Loong than on MEBench, it remains substantial in the **Chain of Reason** task (-5.15 points). This demonstrates that even for tasks framed as "reasoning," delegating logical joins and evidence synthesis to the SQL engine provides a more reliable scaffold than relying solely on the LLM's internal reasoning over massive context.

4. **Error Correction (CLEAR) Ensures Data Integrity**: Disabling the CLEAR mechanism (**w/o CLEAR**) leads to a consistent, measurable drop across all tasks, most noticeably in **Clustering** and **Chain of Reason**. This indicates that enforcing logical consistency during extraction is crucial for maintaining reliable data foundations, which is especially important for the multi-step inferences required in these tasks.

5. **Active Discovery (ASK) Offers Incremental Gains**: The passive schema discovery variant (**w/ Passive Schema Discovery**) performs closest to the full model, yet a clear gap persists, particularly in the **Comparison** task. This suggests that while a strong two-pass discovery method can capture much of the structure, the interactive ASK component provides an important refinement layer, likely by resolving ambiguous entity references critical for accurate comparisons in long texts.

These results robustly demonstrate that each component of DocSage contributes to its state-of-the-art performance on long-document, multi-faceted QA tasks. The structured, agentic approach proves particularly advantageous as document length and task complexity increase.