HiKEY: Hierarchical Multimodal Retrieval for Open-Domain Document Question Answering

Joongmin Shin[1], **Gyuho Shim**[3], **Jeongbae Park**[1], **Jaehyung Seo**[2‡], **Heuiseok Lim**[1,3‡]

[1]Human-inspired AI Research, Korea University
[2]Computer Science and Engineering, Konkuk University
[3]Department of Computer Science and Engineering, Korea University

{tlswndals13, gjshim, insmile, limhseok}@korea.ac.kr
seojae777@konkuk.ac.kr

Abstract

Retrieval-augmented generation (RAG) for document-based Open-domain Question Answering (ODQA) on large-scale industrial corpora faces two critical bottlenecks: routing failure in locating the correct document and evidence fragmentation in integrating scattered information. Existing approaches relying on flat text chunks or page-level images inherently struggle to (i) precisely pinpoint the target document among thousands of candidates and (ii) organically connect multimodal evidence, such as tables and figures, within a limited token budget. To address these challenges, we propose **HiKEY**, a hierarchical tree-based multimodal retrieval framework that elevates document hierarchy to a first-class retrieval signal. Instead of simple chunking, **HiKEY** reconstructs a logical heterogeneous graph via Document Hierarchical Parsing (DHP), explicitly encoding parent–child relationships. Adopting a hierarchical coarse-to-fine strategy, the framework (1) performs global routing to rapidly prune the search space using hierarchical indexing, and (2) conducts fine-grained retrieval to rank sections by employing a multimodal fusion strategy that captures the most discriminative evidence. Finally, **HiKEY** assembles a token-efficient evidence subgraph via a hybrid structural-semantic packing strategy. Experiments on ODQA benchmarks demonstrate that **HiKEY** significantly outperforms page- and chunk-based baselines, improving retrieval recall by up to 12.9% and end-to-end QA performance by up to 6.8%.

1 Introduction

Retrieval-augmented generation (RAG) has become an essential technique for handling large, information-dense document corpora in expert domains such as finance, law, and engineering (Lewis et al., 2021; Jeong, 2023; Ge et al., 2023). In real industrial settings, documents are rarely simple text streams; rather, they typically span dozens of pages (multi-page) and contain a complex mixture of multimodal elements such as tables, figures, schematics, captions, and footnotes (Gong et al., 2020; Qu et al., 2025; Tito et al., 2023). Beyond within-document retrieval, Open-domain Question Answering (ODQA) requires identifying the correct documents and sections from a large corpus. In such settings, system-level performance and cost are determined not only by the generator, but also by the choice of retrieval unit and the efficiency of the exploration procedure (routing and aggregation) (Gao et al., 2024).

The most widely adopted approach is text chunk-based RAG, which splits documents into text chunks and retrieves the top-k chunks by similarity (Gong et al., 2020; Duarte et al., 2024). However, this approach fails to reflect key properties of complex industrial documents. First, the evidence for an answer is often not self-contained within a single paragraph, but fragmented across multiple pages or documents. Second, evidence is entangled with section hierarchies (e.g., 1 $\rightarrow$ 1.2 $\rightarrow$ 1.2.1) and explicit in-text pointers such as "Table 2" or "Figure 3". Third, crucial information is frequently embedded in non-text regions such as tables and charts (Xing et al., 2024b). Most importantly, text chunks are confined to local context and can easily miss global structural signals such as the table of contents or section paths. This often results in (1) routing failure, where the system cannot find the correct document or section during corpus exploration, or (2) incomplete answers due to disconnected evidence (Hong et al., 2024; Shin et al., 2025).

To overcome these limitations, GraphRAG methods that leverage connectivity between

‡Corresponding author.

chunks have been proposed (Sarthi et al., 2024; Liu et al., 2025), but most of them focus on text based links and fail to fully incorporate multimodal elements. As an alternative, multimodal RAG that embeds full-page images has gained attention (Faysse et al., 2025; Cho et al., 2025; Tanaka et al., 2025), but Page-level units inject large amounts of irrelevant content, dramatically increasing token costs, and make it difficult to fit information from many documents within a limited context window (Ma et al., 2025).

We argue that the core to industrial ODQA is not merely expanding modality, but redesigning the retrieval unit and exploration procedure using document hierarchy. Accordingly, motivated by the intuition that hierarchical structure is the key to multimodal ODQA, we propose **HiKEY**, which exploits structural information as a first-class retrieval signal. Specifically, **HiKEY** (1) reconstructs a hierarchical tree from unstructured documents and redefines paragraphs, tables, and figures as multimodal evidence units annotated with section paths, and (2) builds a hierarchical graph that explicitly preserves the logical document structure.

During retrieval, **HiKEY** follows a hierarchical coarse-to-fine strategy, analogous to how humans read long documents: it first uses hierarchical cues to quickly route to a small set of candidate documents, then ranks sections within the routed documents by scoring their multimodal units (text/table/figure) using a combination of lexical, dense text-semantic, and dense visual-semantic signals to capture the most discriminative evidence within the hierarchical context. Finally, **HiKEY** assembles an evidence subgraph under a fixed token budget by employing a hybrid packing strategy that prioritizes sibling nodes and semantic associates, enabling coherent integration of scattered evidence.

Contributions

- **HiKEY**: We propose a hierarchical tree-based multimodal retrieval framework that elevates document hierarchy from passive metadata to a first-class retrieval signal. By reconstructing a logical graph via DHP, **HiKEY** structurally resolves the evidence fragmentation problem inherent in industrial documents, bridging the gap between flat retrieval and complex document layouts.
- **Hierarchical Coarse-to-Fine Retrieval**: We introduce a novel retrieval strategy that integrates global routing with local fine-grained ranking. Specifically, we employ a fine-grained multimodal fusion strategy that identifies the most discriminative evidence within a section, coupled with a graph traversal mechanism that enriches visual units with their hierarchical upper context. This ensures robustness against visual ambiguity and missing captions.
- **Comprehensive Experiments**: Across industrial-aligned multi-page document ODQA benchmarks, **HiKEY** consistently outperforms state-of-the-art text-based and multimodal RAG baselines. Our method improves retrieval performance by 4.5%–12.9% and QA performance by 3.7%–6.8%, validating the efficacy of our ancestry-aware subgraph assembly in token-limited settings.

2 Related Work

Chunking strategies and retrieval units for RAG. RAG performance depends on the retrieval unit and chunking strategy (Gao et al., 2024), ranging from fixed-length segmentation (Gong et al., 2020) to semantic or structure-aware chunking (Duarte et al., 2024; Zhao et al., 2025; Yepes et al., 2024; Verma, 2025). However, text-centric units often miss non-text evidence (tables/figures) and provide weak global structural cues for corpus-level routing in ODQA, even with hierarchical chunking (Shin et al., 2025).

Page-level multimodal retrieval. To compensate for the limitations of text chunking, visual approaches that embed full-page images have been introduced (Faysse et al., 2025; Cho et al., 2025; Tanaka et al., 2025). While they preserve layout information, they suffer from a structural drawback: coarse granularity. First, pages contain substantial irrelevant content that acts as noise and reduces retrieval precision. Second, using full-page images as input in multi-document settings drastically increases token costs and becomes inefficient (Ma et al., 2025). Moreover, simply concatenating pages makes it difficult for models to capture a document's logical flow (section hierarchy) and the explicit links between distant evidence.

Graph-based retrieval. GraphRAG connects information via graphs to support deeper contextual reasoning beyond simple similarity search.

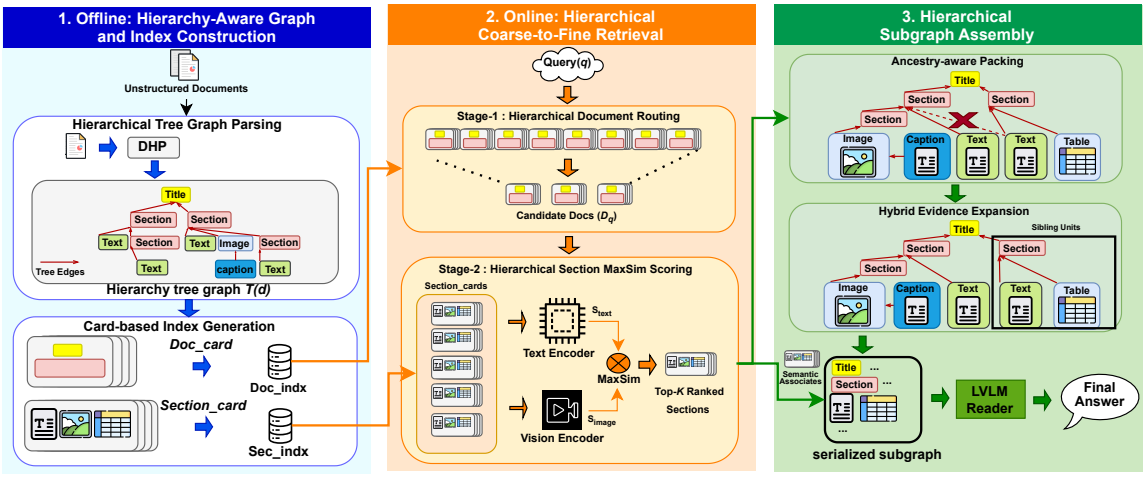

Figure 1: Overview of **HiKEY**, showing 1. offline hierarchy-aware graph and index construction; 2. online hierarchical coarse-to-fine retrieval (Stage-1 document routing followed by Stage-2 section scoring); 3. ancestry-aware evidence subgraph assembly for an LVLM reader.

Prior work extracts entity relations to build knowledge graphs (He et al., 2024) or summarizes and links text chunks for hierarchical exploration (Sarthi et al., 2024; Liu et al., 2025). However, these methods mainly rely on text nodes and cannot use multimodal evidence (tables/figures) as first-class retrieval units. Recent work proposes multimodal graphs with pages as nodes (Wu et al., 2025; Jain et al., 2025), but page-level granularity still limits fine-grained evidence assembly and token efficiency.

Overall, existing approaches have not fully resolved the trade-off between coverage (not missing answer documents) and efficiency (assembling complete evidence under limited resources). To address this, we propose **HiKEY**, which integrates (1) fine-grained multimodal units grounded in DHP-based hierarchy, (2) a hierarchical coarse-to-fine strategy that separates global routing from local ranking, and (3) an ancestry-aware subgraph assembly that dynamically connects scattered evidence via hybrid structural and semantic association.

Industrial document QA and search. Enterprise deployment of Document QA and Search (corpus-level ODQA) is characterized by large repositories of long, multi-page PDFs with mixed modalities (text, tables, figures), complex hierarchies, and limited Reader token budgets – the same operating regime we target in this paper. Prior production-leaning studies focus on individual pieces of this stack: long-context retrieval over enterprise-scale collections (Ma et al., 2025; Shin et al., 2025), layout- and structure-aware chunking for financial/technical documents (Yepes et al., 2024; Verma, 2025; Hong et al., 2024), and multimodal retrieval that preserves page layout (Faysse et al., 2025; Cho et al., 2025; Tanaka et al., 2025). We complement this literature by connecting corpus-level routing, block-level multimodal retrieval, and budget-aware packing into a single framework, and by making the industrial-aligned analyses that matter in deployment first-class evaluation axes: strict routing (*All@K*), evidence-type and multi-hop breakdowns, and token-budget sensitivity (Tables 1, 3, and 4).

Task scope. This paper studies *corpus-level ODQA retrieval over large PDF collections*: **HiKEY** performs hierarchical document routing and section/unit retrieval to assemble evidence under a fixed Reader budget, and we report document identification (Recall/MRR/Hit/All) and end-to-end QA (EM/ANLS/ROUGE-L/METEOR). We do not generate keyphrase lists nor evaluate against keyphrase ground truth (e.g., keyphrase F1@K). While our ODQA benchmarks are open-domain (multi-topic) Wikipedia-style text and layout, our DHP hierarchy backbone is separately validated on scanned/pixel and explicitly multi-domain parsing benchmarks (Appendix D.1, Tab. 11), supporting robustness across document formats and domains.

3 HiKEY

HiKEY is designed to resolve inefficient corpus exploration and evidence fragmentation in ODQA. It is inspired by how humans search long documents: they first skim the table of contents to narrow the scope (routing), and then carefully read the relevant sections and associated tables/figures (fine retrieval). As shown in Fig. 1, the framework consists of two major phases: (1) Offline Hierarchy-Aware Graph and Index Construction, and (2) Online Hierarchical Coarse-to-Fine Retrieval.

Notation. Let $\mathcal{D}$ be a corpus of documents and q a user query. Each document $d \in \mathcal{D}$ is parsed into a hierarchy tree $\mathcal{T}(d)$. We define two types of index cards: (i) a Doc_card x_d representing the global document context, and (ii) a Sec_card x_s containing a set of multimodal units $C_s = \{c_1, c_2, \ldots, c_n\}$ for each section $s \in \mathcal{T}(d)$. Each unit c is assigned a type $\tau(c) \in \{\text{Text, Table, Image}\}$.

3.1 Offline: Hierarchy-Aware Graph and Index Construction

To address the challenge of fragmented evidence highlighted in Sec. 1, we transform each document d from a raw format into a structural heterogeneous graph and a corresponding searchable index.

Hierarchical Tree Graph Parsing. Conventional RAG often loses global context because it splits documents into small, flat chunks. To mitigate this, we use the Document Hierarchical Parsing (DHP) module from M3DocDep (Shin et al., 2026), which is trained on public hierarchy parsing datasets, to reconstruct a logical hierarchy tree $\mathcal{T}(d)$ from page-level layout blocks. Each block (text, table, figure) becomes a node, connected by parent–child Tree Edges (E_{tree}), and receives an ancestor section path (e.g., `Title > Section 5 > 5.3`).

Card-based Index Generation. We generate hierarchical cards for retrieval:

- Doc_card (x_d): title plus section paths for global routing, $x_d = \texttt{Title}(d) \| \bigcup_{s \in \mathcal{T}(d)} \texttt{path}(s)$.
- Sec_card (x_s): a section-level card containing its text and non-text units (tables/figures) as searchable items.

Contextual Enrichment via Graph Traversal. For visual units $c \in \{\text{Table, Image}\}$ lacking captions, **HiKEY** traverses the DHP tree to retrieve a logically preceding textual node as Upper Context, stored as $\text{ctx}(c)$ to reduce visual ambiguity.

3.2 Online: Hierarchical Coarse-to-Fine Retrieval

Given a query q, **HiKEY** identifies the most relevant evidence by traversing the hierarchical search space from the document level down to the multimodal unit level.

Stage-1: Hierarchical Document Routing. To narrow the search scope within a vast corpus, we first identify a candidate set of documents $\hat{\mathcal{D}}_q$ by ranking Doc_cards using a hybrid score:

$$S_{\text{doc}}(d, q) = \alpha \cdot \text{mm}(s_{\text{lex}}(x_d, q)) + (1-\alpha) \cdot \text{mm}(s_{\text{text}}(x_d, q))$$

where $\text{mm}(\cdot)$ denotes per-query min–max normalization. This stage filters out irrelevant documents globally, leveraging the hierarchical paths indexed in the Doc_cards.

Stage-2: Hierarchical Section MaxSim Scoring. Within $\hat{\mathcal{D}}_q$, we perform fine-grained retrieval to rank all constituent Sec_cards globally. We define a Type-Specific Unit Score $s(c, q)$ to leverage modality-specialized encoders:

$$s(c, q) = \begin{cases} s_{\text{hybrid}}(\text{text}(c), q) & \tau(c) = \text{Text} \\ \gamma s_{\text{vis}}(c, q) + (1-\gamma) s_{\text{hybrid}}(\text{ctx}(c), q) & \tau(c) \in \{\text{Table, Image}\} \end{cases}$$

The relevance of a section is determined by the MaxSim operator (Khattab and Zaharia, 2020) over its units: $S_{\text{sec}}(s, q) = \max_{c \in C_s} s(c, q)$. The final section score for ranking is fused as: $S_{\text{final}}(s, q) = \lambda S_{\text{doc}}(\text{doc}(s), q) + (1-\lambda) S_{\text{sec}}(s, q)$.

3.3 Hierarchical Subgraph Assembly

Finally, **HiKEY** assembles the top-K_{sec} ranked sections into a serialized subgraph for the Reader model. To maximize information density within a fixed token budget B_{tok}, we propose a structured assembly process that prioritizes logical coherence and evidence connectivity.

Ancestry-aware Packing. For each selected section, we insert the Anchor Unit (the unit achieving the MaxSim score) and its Governing Headers (ancestral section titles in the DHP tree) to preserve logical coherence and prevent scope misinterpretation.

Hybrid Evidence Expansion. To effectively associate scattered evidence without predicting an explicit cross-block link graph, we employ a hybrid packing strategy to fill the remaining token budget:

- Sibling Units (Structural Association): We first prioritize tables and figures sharing the same parent section as the anchor. This preserves local context and visual-to-text alignment within the same subtree.
- Semantic Associates (Semantic Association): If the budget permits, we further expand the subgraph with non-local visual units that exhibit high vector similarity to the anchor. This captures latent cross-section dependencies, ensuring the Reader can integrate distributed multimodal evidence.

The two expansion modes are complementary: structural association recovers evidence that is guaranteed to be co-referential by document layout (same subtree), while semantic association recovers evidence that the layout fails to link but that is topically aligned with the anchor. An algorithm-aligned schematic of these phases (Fig. 2), a compact running example on the Apollo 11 query, and the exact Reader serialization format are provided in Appendix E, so that the reader can see *what the subgraph contains, how it is constructed, and how it is interpreted*.

4 Experiments

4.1 Experimental Settings

We evaluate **HiKEY** in a realistic corpus-level ODQA setting with a joint index over the entire corpus, where the system must identify relevant documents and sections for each query. Dataset composition, metric definitions, and implementation details are provided in the Appendix.

Datasets. We evaluate on M3DocVQA (Cho et al., 2025) and FRAMES (Krishna et al., 2025), chosen because together they exercise the industrial-ODQA properties identified above: long multi-page PDFs, mixed text/table/figure evidence, fragmented multi-hop evidence, and fixed-budget serving. FRAMES is originally based on Wikipedia text; we re-render it as multi-page PDFs (FRAMES-PDF) to match our setting, yielding an ODQA benchmark with substantial visual and structural complexity. Appendix A.1 spells out how each benchmark maps onto the enterprise requirements listed in the Related Work.

Baselines. We compare **HiKEY** against representative methods spanning four categories: (i) text chunk-based RAG (e.g., LumberChunker (Duarte et al., 2024), MultiDocFusion (Shin et al., 2025)), (ii) text-based GraphRAG (e.g., RAPTOR (Sarthi et al., 2024), HopRAG (Liu et al., 2025)), (iii) page-level multimodal RAG (e.g., ColPali (Faysse et al., 2025)-based M3DocRAG (Cho et al., 2025)), and (iv) page-graph multimodal GraphRAG (e.g., SimpleDoc (Jain et al., 2025)).

Reader and Retrieval Backbones. To isolate the effect of our routing and retrieval framework, all comparisons use the same Reader model and decoding configuration. We employ Qwen2.5-VL-7B-Instruct (Team, 2024) as the Large Vision Language Model(LVLM) Reader and standardize the input context length to 16K tokens across all baselines. This reflects the realistic memory constraint that page-level models can process roughly up to four pages on H100 GPUs (Cho et al., 2025). For retrieval, we utilize BM25 (Robertson and Zaragoza, 2009) for sparse signals; for dense representations, we use gte-Qwen2-7B-instruct (Li et al., 2023) for text and MM-Embed (Sheng-Chieh Lin, 2024) for visual crops (tables/figures). Hyperparameters and library versions are detailed in Appendix F.

Protocol and Evaluation. We evaluate retrieval (Recall, MRR, Hit, All) for document identification and end-to-end QA (EM, ANLS, ROUGE-L, METEOR) for answer quality.

4.2 Experimental Results

We quantitatively analyze how the key design components of **HiKEY** contribute to ODQA performance from five perspectives: (1) corpus-level answer-document localization, (2) robustness under Top-K and token budget constraints, (3) whether retrieval improvements translate to QA quality, (4) handling of table/figure evidence and multi-hop queries, (5) module-level contribution via ablation.

Method	M3DocVQA				FRAMES				AVG			
	Recall	MRR	Hit	All	Recall	MRR	Hit	All	Recall	MRR	Hit	All
Text chunk-based RAG												
Page	80.6	91.3	95.0	66.4	65.4	91.9	97.1	34.4	73.0	91.6	96.1	50.4
Length chunking	82.0	90.6	95.0	69.1	65.3	92.8	96.9	33.8	73.7	91.7	96.0	51.4
LumberChunker	81.5	90.9	95.1	68.1	64.8	92.4	96.8	33.2	73.2	91.7	95.9	50.6
Meta Chunker	81.2	90.8	95.0	67.7	64.9	92.4	96.8	33.3	73.1	91.6	95.9	50.5
Structural chunking	80.6	90.5	94.8	66.9	64.0	92.1	96.6	32.4	72.3	91.3	95.7	49.7
MultiDocFusion	83.0	91.5	95.4	70.5	66.2	93.0	97.2	34.8	74.6	92.2	96.3	52.6
Text-based GraphRAG												
RAPTOR	81.8	90.9	95.0	68.4	64.9	92.0	96.6	33.3	73.4	91.5	95.8	50.9
HopRAG	82.3	91.1	95.2	69.3	65.8	92.2	96.8	34.2	74.1	91.7	96.0	51.8
Page-level multimodal RAG												
M3DocRAG	75.6	81.8	89.9	61.4	56.8	83.2	91.9	25.0	66.2	82.5	90.9	43.2
VDocRAG	77.1	83.3	90.9	63.2	58.5	84.2	92.5	26.5	67.8	83.8	91.7	44.9
Multimodal GraphRAG												
MoLoRAG	76.0	82.4	90.2	62.0	57.4	83.5	92.1	25.5	66.7	83.0	91.2	43.8
SimpleDoc	77.3	83.5	91.1	63.6	60.6	85.4	93.2	28.4	69.0	84.5	92.2	46.0
HiKEY	**84.9**	**91.8**	**96.1**	**73.6**	**73.3**	**94.2**	**97.9**	**46.2**	**79.1**	**93.0**	**97.0**	**59.9**

Table 1: Document-level retrieval results on M3DocVQA and FRAMES, averaged over $K=1,\ldots,10$, reporting Recall, MRR, Hit, and All, plus the dataset average.

4.3 Document Identification: Effect of Stage-1 Routing

Table 1 reports document identification performance (Avg@1–10). **HiKEY** achieves the best results on both datasets, with a particularly large margin on FRAMES, which requires retrieving multiple supporting documents.

M3DocVQA: Stable document localization. **HiKEY** reaches Recall 84.9, surpassing the strong baseline MultiDocFusion (83.0) by +1.9%. On the strict *All* metric, which requires retrieving *all* answer documents, **HiKEY** achieves 73.6, improving by +3.1%. This indicates that our approach of using hierarchy as a first-class retrieval signal is substantially more effective than simple text matching for capturing answer documents.

FRAMES: Resolving the routing bottleneck (Critical Observation). The most notable result appears on FRAMES. Since FRAMES requires retrieving *all* dispersed supporting articles, missing even one document leads to failure. **HiKEY** not only achieves Recall 73.3 (+7.1%), but also reaches 46.2 on *All*, outperforming MultiDocFusion (34.8) by +11.4%.

This suggests that flat/page-level retrieval tends to rely on partial local clues and yields fragmented results, whereas **HiKEY**'s hierarchical routing leverages global section paths to better gather dispersed relevant documents with fewer omissions. In other words, **HiKEY** structurally addresses the chronic ODQA failure mode of routing failure.

Limitations of page-level multimodal retrieval. In contrast, full-page embedding methods (M3DocRAG, VDocRAG) perform worse than text baselines. While page-level representations encode visual information, they lack explicit global structural signals needed to pinpoint relevant documents in large corpora.

4.4 Top-K Sensitivity: High Recall with Few Candidates

Table 5 shows Recall as a function of the number of retrieved documents (K). While **HiKEY** is comparable at R@1, the gap widens rapidly as K increases: **HiKEY** improves by +6.2% at R@5 and +7.6% at R@10.

This indicates that **HiKEY** does not merely "get lucky" with the top-ranked document, but produces higher-quality rankings across the entire Top-K list. The hierarchical coarse-to-fine strategy—stable candidate acquisition via global routing and precise filtering via fine-grained scoring—is highly effective.

4.5 Token Budget Sensitivity: Efficiency under Limited Resources

In practical deployments, the input token budget is limited, making it essential to convey key information efficiently. Table 4 compares document Avg Recall@10 across budgets (0.5K–2K).

HiKEY remains best across all budgets. Even at the very small budget of 0.5K, **HiKEY** achieves 78.8 Avg R@10, outperforming MultiDocFusion (74.3) by +4.5%, and the margin persists as budget increases. This validates the efficacy of our

Method	M3DocVQA				FRAMES				AVG			
	EM	ANLS	ROUGE-L	METEOR	EM	ANLS	ROUGE-L	METEOR	EM	ANLS	ROUGE-L	METEOR
Text chunk-based RAG												
Page	21.1	24.0	25.0	19.0	7.0	9.8	12.7	10.6	14.1	16.9	18.9	14.8
Length chunking	17.5	19.9	21.4	16.7	6.9	9.6	12.9	10.4	12.2	14.8	17.2	13.6
LumberChunker	21.4	24.2	25.6	19.5	6.8	9.4	12.5	10.2	14.1	16.8	19.1	14.9
Meta Chunker	21.3	24.1	25.5	19.4	6.8	9.4	12.5	10.2	14.1	16.8	19.0	14.8
Structural chunking	20.6	23.3	24.6	18.8	6.4	8.9	12.0	9.7	13.5	16.1	18.3	14.3
MultiDocFusion	23.0	25.9	27.3	20.6	7.8	10.8	13.9	11.6	15.4	18.4	20.6	16.1
Text-based GraphRAG												
RAPTOR	22.5	25.3	26.7	20.2	7.5	10.4	13.5	11.2	15.0	17.9	20.1	15.7
HopRAG	22.8	25.7	27.1	20.5	7.7	10.7	13.8	11.5	15.3	18.2	20.5	16.0
Page-level multimodal RAG												
M3DocRAG	24.1	27.0	28.2	21.3	4.2	5.9	9.0	6.7	14.2	16.5	18.6	14.0
VDocRAG	24.4	27.4	28.8	21.6	4.5	6.3	9.4	7.1	14.5	16.8	19.1	14.4
Multimodal GraphRAG												
MoLoRAG	25.2	28.2	30.0	22.4	4.7	6.5	9.6	7.3	15.0	17.4	19.8	14.9
SimpleDoc	25.6	28.7	30.1	22.5	4.9	6.8	9.9	7.6	15.3	17.8	20.0	15.1
HiKEY	**27.5**	**30.7**	**32.2**	**23.9**	**10.5**	**14.6**	**17.7**	**15.4**	**19.0**	**22.7**	**25.0**	**19.7**

Table 2: End-to-end QA performance on M3DocVQA and FRAMES using EM, ANLS, ROUGE-L, and METEOR, plus the dataset average. Higher is better for all metrics.

ancestry-aware packing strategy: by selectively assembling the Anchor Unit and its Governing Headers, **HiKEY** maximizes information density and avoids irrelevant context.

By contrast, page-level models cannot flexibly adapt to token budgets because their input unit (page) is fixed.

4.6 End-to-End QA Performance

To verify whether retrieval improvements translate to answer accuracy, we measure end-to-end QA (Table 2). **HiKEY** achieves consistent improvements in final answer quality.

M3DocVQA. **HiKEY** achieves EM 27.5 and ANLS 30.7, improving over the best baseline (SimpleDoc) by +1.9% and +2.0%, respectively. Even when document identification gaps are modest, QA still improves; this is consistent with **HiKEY**'s fine-grained multimodal fusion selecting more answer-bearing units *within* a retrieved document, but we do not claim this as a conclusion from QA scores alone. The evidence-type and hop breakdown in Sec. 4.7 and the ablations in Sec. 4.9 isolate the specific components responsible.

FRAMES. On FRAMES, **HiKEY** achieves EM 10.5 and ANLS 14.6, improving over MultiDocFusion by +2.7% and +3.8%. The earlier gains in strict document identification (*All*) are reflected in QA performance, highlighting that for composite queries like FRAMES, not missing any supporting document is a key prerequisite for success.

4.7 Analysis by Evidence Type & Hop Count

Industrial documents contain diverse evidence types (text, tables, figures) and frequently require multi-hop aggregation. Table 3 breaks down performance by evidence source and hop count.

Handling non-text evidence (Table/Image). **HiKEY** shows strong gains not only on Text, but also on Table (+2.9%) and Image (+2.8%) queries. This validates our design of treating tables/figures as first-class units enriched with upper context via graph traversal.

Handling multi-hop queries. Performance decreases for all methods as hop count increases, but **HiKEY** best mitigates the degradation. In particular, for challenging 3-hop+ queries, **HiKEY** maintains about 4% higher performance than baselines. This is attributable to our hybrid subgraph assembly that preserves hierarchical context via sibling packing and captures cross-section dependencies via semantic association, preventing fragmentation.

4.8 Why Baselines Fail, Why HiKEY Helps

The consistently strong results above are not a product of a larger budget or a stronger encoder

Method	M3DocVQA								FRAMES			
	Evidence source			Doc-id hop				Overall	Doc-id hop			Overall
	Text	Table	Image	1-hop	2-hop	3-hop	4+	ANLS	2-hop	3-hop	4+	ANLS
Text chunk-based RAG												
Page	30.8	18.5	9.4	25.6	23.8	13.9	15.1	24.0	8.3	11.2	10.0	9.8
Length chunking	30.2	11.5	7.9	16.8	23.1	15.4	23.0	19.9	5.3	7.0	7.3	6.4
LumberChunker	30.9	15.5	8.8	24.8	24.5	15.8	22.6	24.2	8.0	10.8	9.6	9.4
Structural chunking	30.6	16.8	8.6	24.0	23.5	15.0	22.0	23.3	7.5	10.2	9.0	8.9
MultiDocFusion	31.4	19.8	10.8	27.0	25.8	17.2	24.4	25.9	9.2	12.5	11.0	10.8
Text-based GraphRAG												
RAPTOR	31.2	19.0	10.4	26.4	25.1	17.1	23.1	25.3	8.8	12.0	10.5	10.4
HopRAG	31.6	19.7	10.9	26.8	25.5	17.4	23.6	25.7	9.0	12.3	10.9	10.7
Page-level multimodal RAG												
M3DocRAG	31.8	22.0	13.7	31.2	25.2	12.9	2.6	27.0	8.5	7.6	0.3	5.9
VDocRAG	32.2	22.6	14.2	31.8	25.6	13.2	2.8	27.4	9.0	8.1	0.4	6.3
Multimodal GraphRAG												
MoLoRAG	32.7	22.8	14.9	32.7	26.2	11.6	5.0	28.1	9.2	8.3	0.5	6.5
SimpleDoc	33.2	23.6	15.8	33.3	26.8	12.3	6.0	28.7	9.6	8.6	0.6	6.8
HiKEY	**34.1**	**26.5**	**18.6**	**34.0**	**28.8**	**21.3**	**25.6**	**30.7**	**12.8**	**16.3**	**15.0**	**14.6**

Table 3: ANLS breakdown by evidence source (Text/Table/Image) and by multi-hop difficulty (doc-id hops) on M3DocVQA and FRAMES.

Method	0.5K	0.7K	1K	2K	Avg
Text chunk-based RAG					
Page	Fixed	Fixed	Fixed	Fixed	73.3
Length chunking	73.7	73.9	74.1	76.2	74.5
LumberChunker	72.8	73.0	73.2	75.3	73.6
Meta Chunker	72.7	72.9	73.1	75.2	73.5
Structural chunking	72.0	72.2	72.3	74.5	72.8
MultiDocFusion	74.3	74.4	74.6	76.8	75.0
Text-based GraphRAG					
RAPTOR	73.0	73.2	73.4	75.5	73.8
HopRAG	73.7	73.9	74.1	76.2	74.5
Page-level multimodal RAG					
M3DocRAG	Fixed	Fixed	Fixed	Fixed	66.2
VDocRAG	Fixed	Fixed	Fixed	Fixed	67.8
Multimodal GraphRAG					
MoLoRAG	Fixed	Fixed	Fixed	Fixed	66.7
SimpleDoc	Fixed	Fixed	Fixed	Fixed	69.0
HiKEY	**78.8**	**79.0**	**79.1**	**81.3**	**79.5**

Table 4: Token-budget sensitivity comparison averaged over M3DocVQA and FRAMES, measured by budgeted document Recall@10 after evidence packing at token budgets $B_{\text{tok}} \in \{0.5, 0.7, 1, 2\}$K. Page-level methods are marked `Fixed` because their input unit (page) cannot be dynamically resized.

Method	R@1	R@5	R@10	Avg
Text chunk-based RAG				
Page	46.3	74.3	77.3	65.9
Length chunking	47.0	76.6	80.0	67.8
LumberChunker	46.6	76.0	79.4	67.4
Meta Chunker	46.6	75.9	79.3	67.3
Structural chunking	46.1	75.1	78.5	66.6
MultiDocFusion	47.5	77.5	81.0	68.7
Text-based GraphRAG				
RAPTOR	46.8	76.2	79.6	67.5
HopRAG	47.2	77.0	80.4	68.2
Page-level multimodal RAG				
M3DocRAG	41.3	70.3	72.6	61.4
VDocRAG	42.3	72.0	74.4	62.9
Multimodal GraphRAG				
MoLoRAG	41.8	70.9	72.9	61.9
SimpleDoc	43.1	73.3	75.4	63.9
HiKEY	**47.9**	**83.7**	**88.6**	**73.4**

Table 5: Top-K sensitivity averaged over datasets (document recall $R@K$, ↑). Avg is the mean over $K \in \{1, 5, 10\}$.

– all systems are evaluated with the same LVLM Reader, the same corpus split and document renderings/OCR source where applicable, and the same 16K input budget (Sec. 4.1). Rather, each baseline family has a structural limit under corpus-level multimodal ODQA that the corresponding **HiKEY** component is designed to remove. The ablations in Table 6 isolate the hierarchy-indexing, routing, and multimodal-fusion contributions, while the budget-sensitivity results in Table 4 and the qualitative cases in Appendix H support the role of ancestry-aware packing. Table 7 summarizes the mapping between baseline-family limitations, the corresponding **HiKEY** mechanism, and the supporting evidence in our setting.

Appendix H provides qualitative failure cases for each family, and Appendix D.1 reports DHP parser validation and failure-mode analysis that further bound how much of the gain depends on any one component.

Ablation Component	Avg R@10
Stage-1: Hierarchy Indexing Strategy	
Body-only (No hierarchy)	81.2
+ Concat Title/Header	84.6
+ Field-separated Hierarchy (Ours)	88.6
Stage-1: Routing Strategy	
Doc-only routing	87.7
Sec-only routing	76.1
Doc → Sec (Coarse-to-Fine) (Ours)	88.6
Stage-2: Multimodal Fusion Strategy	
Global Search (No routing)	81.0
Candidate Docs Only	87.9
+ Anchor Subtree (Ours)	88.6
BM25 only	87.6
+ Text dense	88.2
+ Visual dense (Full Fusion)	88.6

Table 6: Ablation study measuring averaged R@10 across key HiKEY components (higher is better). We ablate the Stage-1 hierarchy indexing strategy and routing procedure, and analyze Stage-2 design choices including retrieval scope restrictions and multimodal fusion signals.

4.9 Ablation Study

We conduct ablations to analyze the source of **HiKEY**'s gains (Table 6).

Stage-1: Hierarchy Indexing Strategy. Separating hierarchy information (Title/Header) as its own field yields a large gain over naive concatenation (84.6 → 88.6), indicating that hierarchy cues are most effective when used as explicit routing signals.

Stage-1: Routing Strategy. Stepwise routing (Doc → Sec) is most effective compared to Doc-only or Sec-only routing, confirming the benefit of the coarse-to-fine approach.

Stage-2: Multimodal Fusion Strategy. Removing routing and searching corpus-wide causes a sharp drop (81.0), while restricting scope to candidate docs and anchor subtrees recovers performance. Combining lexical, dense text, and visual signals (Full S2) achieves the best result (88.6), confirming the necessity of integrating diverse signals.

5 Conclusion

This work addresses structural bottlenecks in corpus-level ODQA over industrial multi-page documents. We identify that corpus-level routing is a primary failure mode in practice, and that flat chunking further fragments multimodal evidence (tables/figures), hindering multi-hop reasoning.

Baseline family	**HiKEY mechanism (addresses the limitation)**	**Supporting evidence (in our setting)**
Chunk-based text RAG: weak document-level cues for corpus-scale routing; fragmented evidence.	`Doc_card` with a *separate* hierarchy field for Stage-1 routing.	Concat → field-separated hierarchy: 84.6 → 88.6 Avg R@10 (+4.0).
Text-based GraphRAG: tables/figures are not first-class retrieval units.	Sec_card with Type-Specific Unit Scoring; region-level visual embeddings.	Table/Image breakdown (Tab. 3): **HiKEY** improves over the strongest multimodal baselines; adding the visual-dense signal within Stage-2 fusion lifts Avg R@10 from 88.2 to 88.6 (Tab. 6).
Page-level multimodal RAG: whole-page units dilute evidence and saturate the Reader budget.	Block-level units + hierarchical routing; ancestry-aware packing under B_{tok}.	FRAMES strict *All*: MultiDocFusion 34.8 → **HiKEY** 46.2 (+11.4); **HiKEY** best at every budget (0.5K–2K, Tab. 4).
Multimodal page-graph GraphRAG: connectivity without hierarchical scope during serialization.	Ancestry-aware packing with Governing Headers + hybrid sibling/semantic expansion.	Gains hold down to the 0.5K token budget (Sec. 4.5) where page-level units no longer fit.

Table 7: Baseline-family limitations → **HiKEY** mechanism → supporting evidence in our setting. The mapping makes the source of our gains transparent under a controlled, corpus-level multimodal ODQA setting with a fixed Reader budget; these limitations have not been systematically characterized in this setting by prior work.

HiKEY alleviates these issues by leveraging document hierarchy as a first-class retrieval signal.

We propose **HiKEY**, which reconstructs documents into a heterogeneous graph where multimodal blocks are enriched with section paths via DHP. In Stage-1, we employ hierarchical routing to quickly prune the search space, achieving high recall on complex benchmarks like FRAMES. In Stage-2, we perform fine-grained retrieval within the candidates using a multimodal fusion strategy, which significantly improves ranking quality and coverage. Finally, we assemble an ancestry-aware evidence subgraph under a token budget via a hybrid structural-semantic packing strategy, which yields a token-efficient subgraph that preserves hierarchical context for table/figure-heavy queries even under tight budgets.

Experiments on M3DocVQA and FRAMES demonstrate that **HiKEY** consistently improves both document identification and end-to-end QA. Ablations confirm that these gains stem from the synergy between hierarchical indexing, the coarse-to-fine procedure, and multimodal fusion.

Limitations

Preprocessing and Indexing Overhead. Unlike simple text chunking, **HiKEY** requires additional offline computation for DHP-based hierarchy reconstruction and heterogeneous graph construction. While this cost is amortized during inference, future work should explore pipeline optimization and incremental indexing to support environments with frequent document updates.

Dependency on Upstream Modules. Our framework relies on the accuracy of upstream OCR and layout parsing models. Errors in the initial parsing stage can propagate to incorrect section paths and potentially lead to routing failures. Improving robustness against noisy inputs, such as low-quality scans or handwriting, remains an important direction for future research.

Structural Assumptions. **HiKEY** is designed to leverage explicit logical hierarchies (e.g., sections, headers). Consequently, for documents with weak or absent hierarchy—such as flat plain text files, receipts, or unstructured slides—the benefits of our hierarchical routing and ancestry-aware packing strategy may be diminished compared to standard retrieval methods.

Post-retrieval Evidence-State. Our evaluation holds the Reader, OCR pipeline, and 16K token budget fixed, so the gains reported here are retrieval-side improvements under a single delivery configuration. In our setting, retrieval recall improves by up to 12.9% while end-to-end QA improves by 6.8%; closing this gap plausibly requires post-retrieval factors—OCR fidelity, evidence materialization, and reader calibration—that are orthogonal to **HiKEY**'s scope. Consistent with this, standard parser-level hierarchy metrics (F1, STEDS) and downstream retrieval/QA metrics need not move together, so improvements in one should not be extrapolated to the others without controlled evaluation. Systematically separating retrieval gain from reader-side evidence-state factors, and extending beyond a single reader family to weak-hierarchy regimes, is left for future work.

Acknowledgements

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2021R1A6A1A03045425). This work was supported by the Commercialization Promotion Agency for R&D Outcomes (COMPA) grant funded by the Korea government (Ministry of Science and ICT) (2710086166). This work was supported by the Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (RS-2024-00398115, Research on the reliability and coherence of outcomes produced by Generative AI). This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2026-25481153).

References

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Chunsheng Wu, Huajie Tan, Chunyuan Li, Jing Yang, Jie Yu, Xiyao Wang, Bin Qin, Yumeng Wang, Zizhen Yan, Ziyong Feng, Ziwei Liu, Bo Li, and Jiankang Deng. 2025. Llava-onevision-1.5: Fully open framework for democratized multimodal training. *Preprint*, arXiv:2509.23661.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marcal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. 2019. Scene text visual question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4291–4301.

Jaemin Cho, Debanjan Mahata, Ozan Irsoy, Yujie He, and Mohit Bansal. 2025. M3docvqa: Multi-modal multi-page multi-document understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 6178–6188.

André V. Duarte, João DS Marques, Miguel Graça, Miguel Freire, Lei Li, and Arlindo L. Oliveira. 2024. LumberChunker: Long-form narrative document segmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6473–6486, Miami, Florida, USA. Association for Computational Linguistics.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, CELINE HUDELOT, and Pierre Colombo. 2025. Colpali: Efficient document retrieval with vision language models. In *The Thirteenth International Conference on Learning Representations*.

Yair Feldman and Ran El-Yaniv. 2019. Multi-hop paragraph retrieval for open-domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 2296–2309, Florence, Italy. Defines At Least One@k and Potentially Perfect@k (all supporting paragraphs in top-k).

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

J. Ge, Steve Sun, Joseph Owens, Victor Galvez, O. Gologorskaya, Jennifer C Lai, Mark J Pletcher, and Ki Lai. 2023. Development of a liver disease-specific large language model chat interface using retrieval augmented generation. *medRxiv*.

Hongyu Gong, Yelong Shen, Dian Yu, Jianshu Chen, and Dong Yu. 2020. Recurrent chunking mechanisms for long-text machine reading comprehension. pages 6751–6761.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Preprint*, arXiv:2402.07630.

Seongtae Hong, Joong Min Shin, Jaehyung Seo, Taemin Lee, Jeongbae Park, Cho Man Young, Byeongho Choi, and Heuiseok Lim. 2024. Intelligent predictive maintenance RAG framework for power plants: Enhancing QA with StyleDFS and domain specific instruction tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 805–820, Miami, Florida, US. Association for Computational Linguistics.

Chelsi Jain, Yiran Wu, Yifan Zeng, Jiale Liu, Shengyu Dai, Zhenwen Shao, Qingyun Wu, and Huazheng Wang. 2025. SimpleDoc: Multi-Modal document understanding with Dual-Cue page retrieval and iterative refinement. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 28398–28415, Suzhou, China. Association for Computational Linguistics.

CheonSu Jeong. 2023. A study on the implementation of generative ai services using an enterprise data-based llm application architecture. *Adv. Artif. Intell. Mach. Learn.*, 3:1588–1618.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 3948, New York, NY, USA. Association for Computing Machinery.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. 2025. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4745–4759, Albuquerque, New Mexico. 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. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. To-

wards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Hao Liu, Zhengren Wang, Xi Chen, Zhiyu Li, Feiyu Xiong, Qinhan Yu, and Wentao Zhang. 2025. HopRAG: Multi-hop reasoning for logic-aware retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1897–1913, Vienna, Austria. Association for Computational Linguistics.

Jiefeng Ma, Jun Du, Pengfei Hu, Zhenrong Zhang, Jianshu Zhang, Huihui Zhu, and Cong Liu. 2023. Hrdoc: dataset and baseline method toward hierarchical reconstruction of document structures. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Yubo Ma, Jinsong Li, Yuhang Zang, Xiaobao Wu, Xiaoyi Dong, Pan Zhang, Yuhang Cao, Haodong Duan, Jiaqi Wang, Yixin Cao, and Aixin Sun. 2025. Towards storage-efficient visual document retrieval: An empirical study on reducing patch-level embeddings. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19568–19580, Vienna, Austria. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press. See Chapter 8 (Evaluation): recall definition and ranked evaluation via top-k prefixes.

Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar. 2022. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 37433751, New York, NY, USA. Association for Computing Machinery.

Renyi Qu, Ruixuan Tu, and Forrest Sheng Bao. 2025. Is semantic chunking worth the computational cost? In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 2155–2177, Albuquerque, New Mexico. Association for Computational Linguistics.

Johannes Rausch, Octavio Martinez, Fabian Bissig, Ce Zhang, and Stefan Feuerriegel. 2021. Docparser: Hierarchical document structure parsing from renderings. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:4328–4338.

Johannes Rausch, Gentiana Rashiti, Maxim Gusev, Ce Zhang, and Stefan Feuerriegel. 2023. Dsg: An end-to-end document structure generator.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333389.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. *Preprint*, arXiv:2401.18059.

Mohammad Shoeybi Sheng-Chieh Lin, Chankyu Lee. 2024. Mm-embed: Universal multimodal retrieval with multimodal llms. *Preprint*, arXiv:2411.02571.

Joongmin Shin, Chanjun Park, Jeongbae Park, Jaehyung Seo, and Heuiseok Lim. 2025. MultiDocFusion : Hierarchical and multimodal chunking pipeline for enhanced RAG on long industrial documents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20996–21015, Suzhou, China. Association for Computational Linguistics.

Joongmin Shin, Jeongbae Park, Jaehyung Seo, and Heuiseok Lim. 2026. M3DocDep: Multi-modal, multi-page, multi-document dependency chunking with large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Accepted to CVPR 2026; to appear.

R. Smith. 2007. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633.

Ryota Tanaka, Taichi Iki, Taku Hasegawa, Kyosuke Nishida, Kuniko Saito, and Jun Suzuki. 2025. Vdocrag: Retrieval-augmented generation over visually-rich documents. pages 24827–24837.

Qwen Team. 2024. Qwen2.5-vl technical report. *arXiv preprint*.

Rubèn Tito, Dimosthenis Karatzas, and Ernest Valveny. 2023. Hierarchical multimodal transformers for multi-page docvqa. *Preprint*, arXiv:2212.05935.

Prashant Verma. 2025. S2 chunking: A hybrid framework for document segmentation through integrated spatial and semantic analysis. *Preprint*, arXiv:2501.05485.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei

Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. 2025. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *Preprint*, arXiv:2508.18265.

Xixi Wu, Yanchao Tan, Nan Hou, Ruiyang Zhang, and Hong Cheng. 2025. MoLoRAG: Bootstrapping document understanding via multi-modal logic-aware retrieval. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14035–14056, Suzhou, China. Association for Computational Linguistics.

Hangdi Xing, Changxu Cheng, Feiyu Gao, Zirui Shao, Zhi Yu, Jiajun Bu, Qi Zheng, and Cong Yao. 2024a. Dochienet: A large and diverse dataset for document hierarchy parsing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Hangdi Xing, Changxu Cheng, et al. 2024b. Dochienet: A large and diverse dataset for document hierarchy parsing. In *EMNLP*.

Howard Yen, Tianyu Gao, Jinhyuk Lee, and Danqi Chen. 2023. Moqa: Benchmarking multi-type open-domain question answering. In *Proceedings of the Third DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, pages 8–29. Retrieval metrics: accuracy@k (hit@k-style) and MRR@k.

Antonio Jimeno Yepes, Yao You, Jan Milczek, Sebastian Laverde, and Renyu Li. 2024. Financial report chunking for effective retrieval augmented generation. *Preprint*, arXiv:2402.05131.

Jihao Zhao, Zhiyuan Ji, Zhaoxin Fan, Hanyu Wang, Simin Niu, Bo Tang, Feiyu Xiong, and Zhiyu Li. 2025. MoC: Mixtures of text chunking learners for retrieval-augmented generation system. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5172–5189, Vienna, Austria. Association for Computational Linguistics.

Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. 2020. Image-based table recognition: data, model, and evaluation. In *European Conference on Computer Vision (ECCV)*, pages 564–580. Springer.

Appendix

This appendix provides structured details to support reproducibility and verification: (i) datasets and preprocessing, (ii) hierarchical parsing and section-path construction, (iii) hybrid packing for evidence aggregation (Sibling Units + Semantic Associates), (iv) metric definitions and evaluation protocol (including Recall/MRR/Hit/All), (v) baseline configurations, and (vi) runtime and scalability analysis.

A Datasets and Preprocessing Details

All datasets used in this paper are publicly available research benchmarks. For RAG-based QA, we use only publicly released corpora.

A.1 ODQA Datasets

This work follows a corpus-level ODQA setting that reflects real industrial requirements. Unlike standard DocVQA where a single document is provided, corpus-level ODQA jointly indexes a large PDF corpus and requires the system to identify the relevant documents for each question from the entire collection. This allows us to directly measure routing failure, a key bottleneck in industrial RAG applications.

M3DocVQA (Cho et al., 2025). M3DocVQA is an Open-domain document QA benchmark where questions target the entire corpus rather than a specific PDF. Some questions are multi-hop, requiring aggregation across multiple documents, and evidence often resides in non-text elements such as tables and figures. We index the entire M3DocVQA corpus as a single unified retrieval space. *Industrial mapping.* This mirrors enterprise technical-manual and product-datasheet search: a single query is routed against a large PDF repository, and the ground-truth answer is frequently inside a table (specifications) or a figure (schematic) rather than the surrounding prose – which is why evidence-type breakdowns (Tab. 3) and token-budget sensitivity (Tab. 4) are the deployment-relevant measurements.

FRAMES (Krishna et al., 2025). FRAMES is a challenging dataset where answers are dispersed across multiple Wikipedia documents, making routing failure a primary cause of performance drops. To align with our industrial setting, we constructed FRAMES-PDF by converting the original corpus into a multi-page document format. *Industrial mapping.* This mirrors cross-document compliance and due-diligence search: the system must identify and aggregate *every* supporting document (filing, contract, regulatory exhibit) – missing one is an audit failure, which is why the strict *All@K* metric (Tab. 1) is load-bearing here, not just Recall or Hit.

Specifically, we rendered each Wikipedia article into PDF using a Chromium browser and the Playwright API. We preserved vector graphics, hyperlinks, and layout metadata, and ensured that logical objects were not cut at page boundaries. This process produced a corpus with visual and structural complexity comparable to M3DocVQA, serving as a robust testbed for multimodal ODQA.

A.2 Preprocessing Pipeline

To ensure fair comparison, all baselines and **HiKEY** use the same source document renderings and OCR outputs where applicable.

Document Parsing (DP). We employ a layout detection model trained on DocLayNet (Pfitzmann et al., 2022) to identify layout elements (Title, Header, Paragraph, Table, Figure, Caption). We use standard post-processing with a detection confidence threshold ($\tau_{\text{det}} = 0.5$) and NMS IoU threshold ($\tau_{\text{nms}} = 0.5$) to filter noise.

OCR. We use Tesseract (Smith, 2007) to perform OCR on each detected block region independently. Extracted text is lowercased, control characters are removed, and whitespace is normalized before being stored as block metadata.

B Evaluation Metrics & Protocol

This section details the specific metric definitions and evaluation protocols used in our experiments.

B.1 Retrieval Metrics

Let $\mathcal{G}_q$ denote the set of ground-truth answer documents for a query q. We evaluate the quality of the retrieved set TopK(q) using the following four metrics.

Recall@K (Manning et al., 2008). Recall measures the proportion of relevant documents suc-

[1]For non-text units (e.g., figures/tables), **HiKEY** can attach region crops and/or multimodal embeddings, while keeping each unit as a first-class node for retrieval and token-budget packing.

Method	Retrieval Unit	Modality	Field Separation	Graph	Scope
Flat RAG (page, text-only)	Page	Text	Single field (page text)	None	Corpus-level
Flat RAG (chunk, text-only)	Chunk / paragraph	Text	Single field (chunk text)	None	Corpus-level
Text GraphRAG (text-unit graph)	Chunk / paragraph	Text	Single field (text unit)	Text-unit graph	Corpus-level
Page-level multimodal RAG (Faysse et al., 2025; Cho et al., 2025; Tanaka et al., 2025)	Page	Text + Image (page embedding)	Single field (page)	None	Corpus-level
Multimodal page-graph GraphRAG (Wu et al., 2025; Jain et al., 2025)	Page	Text + Image (page embedding)	Single field (page)	Page graph	Corpus-level
HiKEY (tree-indexed multimodal GraphRAG)	Evidence unit (block-level)	Text + Image (region-based)[1]	Field-separated (`Doc_card`, section path, unit text)	Tree-based heterogeneous graph	Corpus-level

Table 8: Method taxonomy along key design axes: retrieval unit granularity, modality, field separation, graph structure, and corpus-level ODQA scope. The table highlights how HiKEY differs via block-level multimodal units and a tree-based heterogeneous graph with explicit fielded routing.

Statistic	M3DocVQA	FRAMES
Documents	3300	2500
Questions	2,441	824
Evidence source counts (with duplicates)		
Text	1,216	–
Table	1,335	–
Image	940	–
Hop distribution (by #doc_id)		
1-hop	1,096 (44.9%)	0 (0.0%)
2-hop	1,193 (48.9%)	311 (37.7%)
3-hop	114 (4.7%)	288 (35.0%)
4+ hop	38 (1.6%)	225 (27.3%)
Hop statistics		
Avg. hop	1.7	3.2
Min hop	1	2
Max hop	8	11

Table 9: Dataset statistics for M3DocVQA and FRAMES. We report the number of samples, evidence-type counts (with duplicates), and hop distributions computed by the number of linked supporting documents (doc_id), along with summary hop statistics.

cessfully retrieved:

$$\text{Recall@K}(q) = \frac{|\mathcal{G}_q \cap \text{TopK}(q)|}{|\mathcal{G}_q|}.$$

MRR@K (Yen et al., 2023). The Mean Reciprocal Rank (MRR) evaluates the ranking quality by considering the position of the first relevant document. Define $\text{rank}_K(d; q)$ as the 1-indexed rank of document d in $\text{TopK}(q)$, setting $\text{rank}_K(d; q) = \infty$ if $d \notin \text{TopK}(q)$. Then:

$$\text{MRR@K}(q) = \frac{1}{\min_{d \in \mathcal{G}_q} \text{rank}_K(d; q)}.$$

Hit@K (Yen et al., 2023). Hit@K simply checks if at least one relevant document is retrieved:

$$\text{Hit@K}(q) = \mathbb{I}\left[\mathcal{G}_q \cap \text{TopK}(q) \neq \emptyset\right].$$

All@K (Feldman and El-Yaniv, 2019). Following the concept of *Potentially Perfect@*K (Feldman and El-Yaniv, 2019), All@K is a strict metric that checks whether *all* ground-truth documents $\mathcal{G}_q$ are contained within $\text{TopK}(q)$. This is critical for multi-hop queries (e.g., FRAMES) where missing a single document leads to failure:

$$\text{All@K}(q) = \mathbb{I}\left[\mathcal{G}_q \subseteq \text{TopK}(q)\right].$$

Reporting Protocol (Avg@1–10). To provide a comprehensive view of ranking performance rather than cherry-picking a specific K, all document retrieval results in Table 1 are reported as the average of the metric values calculated at each cut-off $K \in \{1, \ldots, 10\}$.

B.2 QA Metrics

Answer quality is evaluated after standard normalization (lowercasing, punctuation removal). We use EM (Exact Match), ANLS (Biten et al., 2019), ROUGE-L (Lin, 2004), and METEOR (Banerjee and Lavie, 2005). For ANLS, we follow the threshold-based implementation standard in MP-DocVQA (Tito et al., 2023).

B.3 Token Budget Protocol

Budget Constraint. For fairness, we strictly control the input context size. We include retrieved evidence units in their ranking order until the cumulative serialized token count reaches the budget B_{tok}.

Comparison with Page-level Models. Page-embedding methods (e.g., ColPali) cannot dynamically adjust their unit size because the retrieval unit is fixed to a full page image. Consequently, they are marked as *Fixed* in our budget-sensitivity analyses and are compared separately under a constraint of equivalent page counts (typically 4 pages $\approx$ 16K tokens).

C Baseline Implementation

For fair comparison, all baselines use the same source corpus, corpus split, Reader, and decoding settings. Each method constructs its own retrieval index according to its retrieval unit and modality assumptions. Differences arise primarily from retrieval unit choice, routing strategy, and modality utilization.

C.1 Text Chunk-based RAG

The most common RAG setup: flat retrieval over text units without hierarchical routing.

Page (Text-only). Treats an entire page's extracted text as a single retrieval unit.

Length Chunking (Gong et al., 2020). Mechanically splits text by fixed length (tokens/chars), often disrupting context.

LumberChunker (Duarte et al., 2024). Uses an LLM to detect topic shifts and set chunk boundaries dynamically based on semantic coherence.

Meta Chunker (Zhao et al., 2025). Merges chunks at the sentence/paragraph level by analyzing perplexity distributions.

Structural chunking (Yepes et al., 2024). Chunks based on layout types (headers, paragraphs), but unlike **HiKEY**, it does not reconstruct a full section tree or treat tables/figures as first-class retrieval nodes with ancestry-aware packing.

MultiDocFusion (Shin et al., 2025). A strong hierarchy-aware baseline that preserves some structure, but lacks field-separated routing and multimodal unit fusion.

C.2 Text-based GraphRAG

Connects text units via graphs to enable multi-step exploration, but relies solely on text.

RAPTOR (Sarthi et al., 2024). Clusters and summarizes chunks to form a tree. Note that this *induced* tree is semantic and not tied to the document's native layout structure.

HopRAG (Liu et al., 2025). Expands search along a similarity graph of text units using LLM-generated pseudo-queries, but does not natively handle tables or figures.

C.3 Page-level Multimodal RAG

Retrieves by embedding full-page images. While capturing layout, they suffer from coarse granularity.

M3DocRAG & VDocRAG. M3DocRAG (Cho et al., 2025) and VDocRAG (Tanaka et al., 2025) both utilize multimodal embeddings but differ in retrieval formulation. M3DocRAG employs a ColPali-style late-interaction retriever (token-level page embeddings scored by MaxSim). We adapt this to the ODQA setting by flattening all pages across the corpus into a single joint index. In contrast, VDocRAG uses an LVLM-based dual-encoder that compresses each page image into a single dense representation (e.g., via EOS-token pooling) for maximum inner-product search.

C.4 Multimodal GraphRAG

Constructs graphs where nodes are full pages, limiting fine-grained control.

MoLoRAG (Wu et al., 2025) & SimpleDoc (Jain et al., 2025). These methods expand retrieval via page connectivity (e.g., similarity edges). However, the page-level granularity limits their ability to assemble a precise evidence subgraph under tight token budgets.

D Hierarchical Parsing, Section Path, and `Doc_card`

This section details the offline construction process of **HiKEY**: hierarchical parsing (DHP), section-path extraction, and the practical construction of `Doc_card`.

D.1 Document Hierarchical Parsing (DHP)

Role in HiKEY. HiKEY relies on an upstream Document Hierarchical Parsing (DHP) module to recover document structure from raw PDFs. Given a document d, DHP produces (i) a hierarchy tree

Method	Unit	Retriever Modality	Hierarchy Signal	Stage-1 Routing	Stage-2 Scope	Evidence Structure	Packing under B_{tok}
Text chunk-based RAG							
Page (Text-only)	Page	Text (OCR)	None	N	Corpus-wide	None	Fixed top-K pages (page-budget)
Length chunking (Gong et al., 2020)	Chunk	Text (OCR)	None	N	Corpus-wide	None	Greedy top-k chunks
LumberChunker (Duarte et al., 2024)	Chunk	Text (OCR)	None	N	Corpus-wide	None	Greedy top-k chunks
Meta Chunker (Zhao et al., 2025)	Chunk	Text (OCR)	None	N	Corpus-wide	None	Greedy top-k chunks
Structural chunking (Yepes et al., 2024)	Chunk	Text (OCR)	Weak (layout-based boundaries)	N	Corpus-wide	None	Greedy top-k chunks
MultiDocFusion (Shin et al., 2025)	Hier. chunk	Text (OCR)	Native / structure-aware	N (no `Hierarchy Field`)	Corpus-wide	No explicit graph (chunk hierarchy only)	Greedy top-k chunks (no sub-graph assembly)
Text-based GraphRAG							
RAPTOR (Sarthi et al., 2024)	Chunk + Summary	Text	*Induced* (summary tree)	N	Corpus-wide	Recursive summary tree	Greedy top-k nodes
HopRAG (Liu et al., 2025)	Chunk	Text	None	N	Corpus-wide	Chunk graph (similarity / hop expansion)	Greedy top-k chunks
Page-level multimodal RAG							
M3DocRAG (Cho et al., 2025)	Page	Page-level MM	None	N	Corpus-wide	None	Fixed top-K pages (page-budget)
VDocRAG (Tanaka et al., 2025)	Page	Page-level MM	None	N	Corpus-wide	None	Fixed top-K pages (page-budget)
Multimodal GraphRAG							
MoLoRAG (Wu et al., 2025)	Page	Page-level MM	None	N	Corpus-wide	Page graph expansion	Fixed top-K pages (page-budget)
SimpleDoc (Jain et al., 2025)	Page	Page-level MM	None	N	Corpus-wide	Page-level graph	Fixed top-K pages (page-budget)
HiKEY	Fine-grained MM unit	Text + Visual Crop	Native (DHP path)	Y: Hierarchy Routing	Cand. docs + Anchor sub-tree	Heterogeneous Tree Graph (tree edges only)	Ancestry-aware hybrid packing (Sibling + Semantic)

Table 10: Qualitative comparison of retrieval framework design choices. We contrast methods by retrieval unit, retriever/modality, hierarchy usage and Stage-1 routing, Stage-2 search scope, evidence structure (none vs. graph), and packing under a token budget B_{tok}.

Method	**HRDoc-S F1/STEDS**	**HRDoc-H F1/STEDS**	**DocHieNet F1/STEDS**	**Avg F1/STEDS**
Generic LVLM baselines (zero-shot image understanding)				
LLaVA-OneVision1.5 (An et al., 2025)	27.61/12.93	26.30/18.21	17.78/ 8.57	23.90/13.24
InternVL3.5 (Wang et al., 2025)	28.40/14.47	27.57/19.98	18.18/ 9.60	24.72/14.68
Qwen2.5-VL (Team, 2024)	28.41/14.51	27.57/19.99	18.20/ 9.62	24.73/14.71
Structured hierarchy parsers (trained for tree reconstruction)				
DocParser (Rausch et al., 2021)	47.09/31.03	35.41/27.15	10.68/ 4.31	31.06/20.83
DSG (Rausch et al., 2023)	48.43/32.13	36.42/27.69	26.71/19.45	37.19/26.42
DSPS (Ma et al., 2023)	65.27/59.57	54.06/38.41	35.61/23.81	51.65/40.60
DSHP-LLM (Shin et al., 2025)	44.90/29.52	61.29/51.34	64.29/53.49	56.83/44.78
Qwen2.5-VL–DHP–SFT	50.97/46.75	43.05/41.02	42.85/40.39	45.62/42.72
M3DocDep (DHP used in HiKEY)	**82.87/76.52**	**77.75/71.65**	**76.01/70.83**	**78.88/72.99**

Table 11: Standalone validation of the DHP backbone used in HiKEY, adapted from M3DocDep (Shin et al., 2026). We use the same DHP checkpoint as the upstream hierarchy parser in HiKEY, and report F1 and STEDS on HRDoc-S, HRDoc-H, and DocHieNet.

$\mathcal{T}(d)$ over layout blocks (e.g., headings, paragraphs, captions, tables, and figures), which provides reliable section paths for Stage-1 routing and ancestry context for evidence packing. Concretely, the recovered section path is serialized into the hierarchy field of each Doc_card/Sec_card, enabling query routing by document-level structure rather than flat chunk similarity.

LVLM-based multimodal DHP. A key distinction from prior DHP-style components is that our DHP is an *LVLM-based multimodal* parser: it jointly consumes text, table crops, and figure crops as first-class inputs to hierarchy recovery. By contrast, the closest prior components are typically LLM-based text-only parsers that cannot leverage visual signals (e.g., a table whose structure is visible only from its rendered layout, or a figure whose caption is laid out across columns). This design choice is load-bearing for **HiKEY**: because downstream retrieval units include tables and figures as first-class nodes, the hierarchy over those units has to be recovered from a multimodal, not text-only, signal.

Model and outputs. We implement DHP with a structure-aware parser that operates on page-level layout blocks and predicts parent–child relations to reconstruct $\mathcal{T}(d)$, while also identifying block types and span boundaries required by downstream indexing. All retrieval experiments in this paper use the same fixed DHP checkpoint; DHP is not tuned on ODQA benchmarks, ensuring that downstream gains are attributable to the retrieval framework rather than task-specific retraining of the parser.

Standalone validation of DHP reliability. A key concern is whether hierarchy recovery is ac-

curate enough to support routing and structured packing. We therefore include component-level validation results for the same DHP checkpoint used in HiKEY, adapted from M3DocDep (Shin et al., 2026). Appendix Table 11 reports F1 and STEDS (Zhong et al., 2020) on HRDoc-S, HRDoc-H (Ma et al., 2023), and DocHieNet (Xing et al., 2024a). These results are included only to characterize the reliability of the upstream parser; all ODQA retrieval and QA results in this paper are evaluated within the HiKEY pipeline.

Downstream sensitivity. Beyond standalone parsing scores, we additionally quantify the importance of hierarchy signals for retrieval. Our ablation study (Table 6) shows that removing hierarchy-aware indexing/routing substantially degrades retrieval quality compared to field-separated hierarchy indexing with coarse-to-fine routing. This indicates that HiKEY's improvements are not merely due to larger contexts or stronger encoders, but critically depend on accurate structural signals supplied by DHP.

Representative DHP failure cases and downstream impact. We catalog two failure modes that dominated the residual errors on HRDoc-S/H and DocHieNet and we trace their effect on HiKEY's retrieval and QA. (**F-H**) *Missing or misclassified headers.* When a header block is detected as a paragraph (or vice versa), the affected unit's Governing Header defaults to a higher ancestor (or the document Title), which widens the section path. Downstream, this hurts Stage-1 routing in two ways: the `Doc_card` hierarchy field loses a discriminative section string, and the affected Sec_card is merged with its neighbors, reducing Stage-2 ranking precision. In our error inspection, documents with frequent header drops tended to show lower Recall@1 than documents with comparable body-text retrieval quality. (**F-N**) *Incorrect nesting (wrong parent).* When a deeper header is attached to the wrong ancestor (e.g., a subsection nested under a sibling section), the resulting section path is structurally plausible but semantically mis-scoped. This does not noticeably affect retrieval (the routing signal is coarse), but it harms Reader interpretability: during Phase 1 packing, the Anchor Unit is shipped with a path that points to the wrong section, which can mislead the Reader on queries that depend on scope ("under which section does X fall?"). Ancestry-aware packing mitigates this partially by always attaching the full path; the Reader at least sees the candidate ancestor chain. Both failure modes are observable in our standalone parsing metrics (STEDS captures nesting errors; F1 captures header-type errors), which is why we retain both numbers in Table 11 rather than reporting a single aggregate.

DHP is an offline, one-time indexing cost. We stress that DHP is executed once per document during offline indexing, not at query time. Once $\mathcal{T}(d)$ and the section paths are computed, they are persisted in the index; every subsequent query reuses them without re-running DHP. The end-to-end latency a deployed HiKEY system exposes at serving time is therefore bounded by Stage-1 routing + Stage-2 scoring + Reader inference, none of which invoke DHP. The runtime breakdown in Appendix I (Table 13) reports this offline cost explicitly, separated from query-time latency; the per-document DHP cost amortizes over the query stream in any realistic deployment.

Discussion. DHP is an upstream module and may be imperfect on severely degraded scans or atypical layouts; however, HiKEY is designed to be robust by combining hierarchy-based routing with multimodal fine retrieval and budgeted evidence packing. The standalone validation in Table 11 bounds the parser's average accuracy, the downstream ablations in Table 6 isolate the contribution of the hierarchy signal within the pipeline, and the offline cost argument above addresses deployment efficiency. Together, these address concerns about the reliability and practical cost of hierarchy parsing in our retrieval pipeline.

D.2 Section-path Extraction

Governing Header. For each evidence unit (block) c, we traverse upward in the recovered tree $\mathcal{T}(d)$ and select the nearest `Title` or `Section Header` ancestor as its *governing header*.

Section Path. Using the header sequence from the document root to the governing header, we construct the section path:

$$\texttt{section\_path}(c) = \texttt{Title} > \texttt{Sec} > \texttt{Subsec} > \ldots$$

This path is stored as structural metadata and is utilized as a key feature for both the hierarchy field index and graph traversal.

D.3 `Doc_card` Construction

Stage-1 routing relies on a lightweight summary representation, `Doc_card`(d), designed to pro-

vide global topic cues without the noise of the full body text.

Deterministic Construction Rules. To implement the theoretical definition in Sec. 3 efficiently, we construct the `Doc_card` by concatenating: (i) The detected `Title` text (selected via DP confidence and position), (ii) A reading-order list of high-level section headers (typically depth 1–2), (iii) Optionally, ToC entries if a Table of Contents page is identifiable. For indexing stability, the final string is truncated to a fixed maximum length (tuned on the validation set).

Design Intent. This construction keeps the Stage-1 index compact while supplying strong global topic signals, preventing the routing module from being overly sensitive to noisy body OCR or irrelevant local details.

D.4 No Explicit Cross-Block Link Modeling in HiKEY

HiKEY does not model, predict, or construct any explicit cross-block link graph beyond the hierarchy tree recovered by DHP. In our pipeline, DHP is used solely to reconstruct the parent–child hierarchy $\mathcal{T}(d)$ (tree edges) and to provide ancestry context (section paths and governing headers). All cross-unit association needed for ODQA is handled deterministically at packing time via the hybrid policy described below.

D.5 Semantic Associate Mining for Hybrid Packing

For each Stage-2 anchor unit c_i, we mine Semantic Associates by ranking other units within the same document using a similarity function $Sim(c_i, \cdot)$ computed from precomputed dense embeddings: text embeddings for textual units, and visual embeddings for table/figure crops. During packing, we add high-similarity visual units as Semantic Associates subject to the token budget, and attach the required ancestry context from the DHP tree to keep each added unit interpretable in its original section context.

E Ancestry-aware Evidence Subgraph Assembly

This section details the ancestry-aware packing policy used to assemble the final multimodal context under a strict token budget B_{tok}.

E.1 Serialization Format

To ensure the Reader model perceives the logical structure, each selected evidence unit is serialized with its structural context:

- Ancestry Context: The document Title and the sequence of governing headers (from the DHP tree) required to locate the unit (e.g., `# 2. Methods > ## 2.1. Model`).
- Unit Metadata: Unique identifiers, unit type (Text/Table/Figure), and source page number.
- Content: The unit's textual content (OCR text, linearized table, or caption).
- Visual Crop: For `Table` and `Figure` units, the corresponding image crop is inserted (if the LVLM budget allows).

E.2 Ancestry-aware Packing Algorithm

Unlike naive greedy packing, our strategy explicitly prioritizes the structural integrity of evidence. We include high-scoring Stage-2 units first, mandatorily attach their ancestry nodes, and then expand using (i) Sibling Units under the same parent section and (ii) Semantic Associates selected by embedding similarity, to form a coherent subgraph under the token budget.

What does the "subgraph" contain? Each assembled evidence subgraph $\mathcal{S}$ is a set of nodes drawn from the DHP tree $\mathcal{T}(d)$ together with their ancestry chains, grouped into three disjoint roles: (1) one or more *Anchor Units* (the highest-ranked Stage-2 units for the query); (2) the *Governing Headers* of each included unit (the chain of ancestral section titles from the document root); (3) optional *Sibling Units* under the same parent section as an Anchor (Phase 2) and *Semantic Associates* retrieved from elsewhere in the document by embedding similarity to an Anchor (Phase 3). The subgraph is *not* a newly learned cross-block link structure: edges are exactly the parent–child tree edges of $\mathcal{T}(d)$ restricted to the included nodes, with no predicted cross-unit links (Appendix D.4).

Schematic of the packing phases. Figure 2 visualizes the three phases of Algorithm 1 for a single Stage-2 anchor. Phase 1 seats the Anchor Unit together with its Governing Headers. Phase 2 draws Sibling Units from the same parent subtree (structural association). Phase 3 draws Semantic Associates from elsewhere in the document

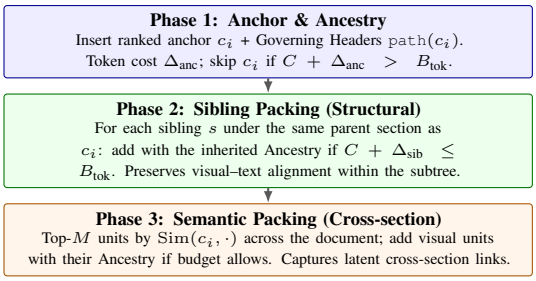

Figure 2: Schematic of the three packing phases in Algorithm 1. Each phase is budget-gated; the loop iterates over ranked Stage-2 anchors.

via embedding similarity (semantic association). Each phase is gated by the remaining token budget $B_{\text{tok}} - C$; any unit that would overflow is skipped and the loop advances to the next ranked anchor.

Compact running example. We instantiate the three phases on the Apollo 11 query from Appendix G (*"Who were the crew members of Apollo 11, and when did they land on the Moon?"*). Assume the top-ranked Stage-2 anchor is the crew paragraph c_1 (Text) under section `3.1 Prime crew`.

- *Phase 1 (Anchor + Ancestry)* seats c_1 together with its Governing Headers `Apollo 11 > 3 Mission personnel > 3.1 Prime crew` ($\Delta_{\text{anc}} \approx 180$ tokens under the Qwen2.5-VL tokenizer).
- *Phase 2 (Sibling expansion)* adds the crew table c_2 and the crew portrait figure c_3; both are Sibling Units under `3.1 Prime crew` (each $\Delta_{\text{sib}} \approx 350$–$450$ tokens including vision-encoder patches).
- *Phase 3 (Semantic expansion)* detects that the landing-time table under `5.1 Landing` has high visual–semantic similarity to c_1 (the query also asks about "landing"); it is added as a Semantic Associate, and its own Governing Headers `5 Mission events > 5.1 Landing` are attached so the Reader can locate it.

The assembled subgraph is thus $\mathcal{S} = \{c_1, c_2, c_3, c_{\text{landing}}\}$ together with two ancestry chains, and is serialized into the Reader prompt as shown in Appendix G. Critically, the Reader sees each unit together with its own ancestry, so a cross-section unit added via Phase 3 does not cause scope confusion – it is explicitly tagged with the section path `5.1 Landing`, distinct from the anchor's path `3.1 Prime crew`.

Algorithm 1 An ancestry-aware hybrid packing algorithm that assembles a coherent evidence subgraph using Sibling and Semantic strategies.

Require: Stage-2 ranked anchors $\{(c_i, s_i)\}_{i=1}^{N}$, DHP tree $\mathcal{T}(d)$, semantic similarity function $Sim(\cdot, \cdot)$, token budget B_{tok}
Ensure: Evidence subgraph $\mathcal{S}$

$\mathcal{S} \leftarrow \emptyset$, $C \leftarrow 0$ ▷ C: current accumulated token count
for $i = 1$ to N **do**
  Phase 1: Anchor & Ancestry
  $\Delta_{\text{anc}} \leftarrow$ tokens for unit c_i + Ancestry Context
  if $C + \Delta_{\text{anc}} > B_{\text{tok}}$ **then** continue
  end if
  Add c_i and Ancestry to $\mathcal{S}$; $C \leftarrow C + \Delta_{\text{anc}}$
  Phase 2: Sibling Packing (Structural)
  $Siblings \leftarrow$ {units sharing same parent section with c_i}
  for $s \in Siblings$ **do**
    $\Delta_{\text{sib}} \leftarrow$ tokens for s ▷ Sibling s inherits the same Ancestry Context as c_i
    if $C + \Delta_{\text{sib}} \leq B_{\text{tok}}$ and $s \notin \mathcal{S}$ **then**
      Add s (with inherited Ancestry) to $\mathcal{S}$; $C \leftarrow C + \Delta_{\text{sib}}$
    end if
  end for
  Phase 3: Semantic Packing (Cross-Section)
  $Candidates \leftarrow$ Top-M units by $Sim(c_i, \cdot)$ from Doc
  for $m \in Candidates$ **do**
    if m is visual unit and $m \notin \mathcal{S}$ **then**
      $\Delta_{\text{sem}} \leftarrow$ tokens for m + Ancestry
      if $C + \Delta_{\text{sem}} \leq B_{\text{tok}}$ **then**
        Add m and Ancestry to $\mathcal{S}$; $C \leftarrow C + \Delta_{\text{sem}}$
      end if
    end if
  end for
end for
return $\mathcal{S}$

Implementation Note. Token counting is performed using the specific tokenizer of the Reader model (Qwen2.5-VL). Image crops are assigned a fixed token cost (corresponding to the vision encoder's patch tokens), and we strictly cap the total number of images to satisfy the LVLM's maximum input resolution constraints.

F LVLM and RAG Settings

This section provides detailed experimental settings (retriever, reader, and hyperparameters) to ensure reproducibility. All experiments are implemented using PyTorch and HuggingFace Transformers.

F.1 Retrieval Configuration

Sparse Retrieval. We use standard BM25 settings from production search libraries (e.g., Pyserini/Elasticsearch). After stopword removal, we use tokenized terms (e.g., from a morphological

Setting	Configuration
Infrastructure	NVIDIA H100 (80GB) × 4
Index Unit	Corpus-level Joint Indexing
Sparse Retriever	BM25 ($k_1 = 1.5, b = 0.75$)
Dense Retriever (Text)	`gte-Qwen2-7b-instruct`
Dense Retriever (Visual)	`MM-Embed`
Strategy	Stage-1: Hierarchical Routing (BM25 + Dense) Stage-2: Fine-grained Multimodal Fusion
Reader Model	`Qwen2.5-VL-7B-Instruct`
Context Limit	16,384 tokens
Decoding	Greedy decoding (Temperature=0.0)
Evaluation Metrics	Retrieval: Recall@K, MRR@K, Hit@K, All@K QA: EM, ANLS, ROUGE-L, METEOR

Table 12: Experimental configuration summary for retrieval and QA. We list infrastructure, sparse/dense retrievers (text and visual), the two-stage strategy (hierarchical routing + fine-grained fusion), the LVLM reader configuration, and evaluation metrics.

analyzer) for both Stage-1 routing fields and Stage-2 text units.

Dense Retrieval. We employ gte-Qwen2-7b-instruct (Li et al., 2023) for text embeddings with an 8192-token window to handle long contexts. For visual evidence units (tables/figures), we crop the regions and encode them using `MM-Embed` (Sheng-Chieh Lin, 2024) into 4096-dimensional embeddings. Signals are combined via late fusion using learned weights tuned on the validation set.

F.2 Reader Configuration

Model & Decoding. We use Qwen2.5-VL-7B-Instruct as the Reader. For consistency and reproducibility, we fix Temperature to 0.0 (greedy decoding) and cap maximum generation at 256 tokens.

Input Context. The retrieved ancestry-aware evidence subgraph is serialized into a textual prompt structure. The total input length is strictly limited to 16K tokens to reflect realistic H100 memory constraints.

F.3 Prompt Template

The QA prompt template used in our experiments is illustrated in Fig. 3. We strictly instruct the model to rely solely on the provided `[Context]` (the assembled subgraph) to minimize hallucinations.

QA Prompt Template

[System Instruction]
You are an AI assistant that answers questions by analyzing the provided documents.
Write an accurate answer to `[Question]` using only the `[Context]` given below.
- Do not answer using information that is not present in the context.
- When referring to tables or figures, explicitly mention their IDs (e.g., Figure 3).
- If you cannot be confident, output "I do not have enough information to answer."

[Question]
`<QUESTION_TEXT>`

[Context]
`<SERIALIZED_EVIDENCE_SUBGRAPH>`

[Answer]

Figure 3: The LVLM QA prompt template, where the [Context] slot is filled with the serialized ancestry-aware evidence subgraph assembled under the token budget.

G Qualitative Case Study and Analysis

This section analyzes how **HiKEY** processes Wikipedia-style multimodal documents. We trace how the pipeline in Fig. 1 applies to an *"Apollo 11"* document (step-by-step walkthrough) and visualize the final evidence subgraph serialized for the LLM Reader.

G.1 System Walkthrough

When a user asks: *"Who were the crew members of Apollo 11, and when did they land on the Moon?"*, **HiKEY** generates an answer via the following internal steps.

Inference Walkthrough: Apollo 11 Mission Query

(a) Document Parsing & OCR.
We parse `Apollo_11.pdf` and extract layout blocks.
- `B_h`: Heading "3. Mission personnel" / "5.1 Landing"
- `B_p`: Text "The prime crew selected for Apollo 11 consisted of..."
- `B_t`: Table "Position | Astronaut" (Commander: Neil Armstrong...)
- `B_f`: Figure (Official Crew Portrait) (+ Crop)

(b) Hierarchy & Graph Construction.
The DHP model reconstructs the ToC-like structure and assigns section paths.
- Path: `Apollo 11` →

3 Mission personnel → 3.1 Prime crew

- Units: c_1(Text), c_2(Table), c_3(Figure)
- Edges: Tree(Sec 3.1 → c_1, c_2, c_3)

(c) Stage-1: Hierarchical Routing.
We match query keywords ("crew", "landing") against the hierarchy field (Doc_card) to narrow the scope.

- Output: Target Doc={Apollo_11.pdf}, Anchors={Sec 3.1, Sec 5.1}

(d) Stage-2: Fine Retrieval & Assembly.
We retrieve units under the anchors and perform ancestry-aware packing of the text (c_1), the crew table (c_2), and the landing-time table, strictly adhering to the token budget B_{tok}. The crew table (c_2) is included as a Sibling Unit, and the landing-time table is retrieved as a Semantic Associate.

G.2 Serialized Input Visualization

The selected evidence units are serialized and fed to the LVLM. The example below illustrates the coherent integration of textual explanation with the crew table and a linked crew photo.

Example of Serialized Evidence Subgraph (Input Prompt)

```
# Global Context
[DOC_META]
ID: Apollo_11.pdf | Title: Apollo 11 - Wikipedia
```

```
# Hierarchy Anchor (Routing Result)
[SCOPE] 3. Mission personnel > 3.1. Prime crew

[UNIT id=42 | Type=Text]
Path: 3. Mission personnel > 3.1. Prime crew
Content: The prime crew for Apollo 11 consisted of Commander Neil Armstrong, Command Module Pilot Michael Collins, and Lunar Module Pilot Edwin "Buzz" Aldrin. Their positions are detailed in Table 1.

[UNIT id=43 | Type=Table]
Path: 3. Mission personnel > 3.1. Prime crew
Visual: <|image_token_1|> (Table Crop)
```

[IMAGE PLACEHOLDER]
(Cropped image of Table 1: Prime Crew Members)

⇓ *Sibling Evidence (Same Section)* ⇓

```
[UNIT id=45 | Type=Figure]
Path: 3. Mission personnel > 3.1. Prime crew
Caption: Figure 2. The official crew portrait of Apollo 11.
Visual: <|image_token_2|> (Photo Crop)
```

[IMAGE PLACEHOLDER]
(Cropped photo of Armstrong, Collins, and Aldrin)

H Qualitative Failure Cases of Baseline Families

We summarize three representative failure modes repeatedly observed when inspecting baseline outputs on FRAMES and M3DocVQA. These are intended to make concrete the structural limits captured by Table 7 in the main text; we do not claim they are exhaustive.

(F1) Chunk-RAG: routing locked onto a lexically similar but topically wrong document. For composite FRAMES queries that share named entities across unrelated articles (e.g., several Wikipedia pages that mention "Apollo"), flat chunk retrievers rank chunks from a nearby but incorrect document at the top because the similarity signal is dominated by local token overlap. Once the wrong document is selected, subsequent chunks reinforce the mistake, which is one reason chunk-RAG's FRAMES *All* score is substantially below its Hit score. **HiKEY** avoids this by routing at the Doc_card level, where the hierarchy field (section paths) separates topically different documents even when their body text looks similar.

(F2) Page-level multimodal RAG: correct page retrieved, answer drowned by page-wide noise. Page-embedding methods often retrieve the correct page but then fail in end-to-end QA because the fixed page unit injects large amounts of unrelated text and visual noise into the 16K budget, pushing the actual answer span below the model's attention priority. This is consistent with the gap between their retrieval Hit@K and end-to-end EM in Tables 1 and 2. **HiKEY** instead packs a block-level Anchor Unit with only the ancestry needed for interpretation, preserving budget for sibling/semantic expansion.

(F3) Text-only GraphRAG: multi-hop succeeds on text but breaks on table/figure-anchored evidence. RAPTOR and HopRAG connect text chunks but do not expose tables and figures as first-class retrieval targets; for M3DocVQA questions whose answer lives in a table row or a figure caption, these systems at best retrieve a nearby text chunk that *mentions* the table, missing the content itself. **HiKEY**'s Sec_card treats the table/figure crop as the retrieval unit with Upper Context attached, so the answer-bearing unit is ranked and packed directly (see Tab. 3 for the Table/Image breakdown).

Stage	Latency (s)			Complexity
	5 pages	10 pages	20 pages	
Visual Analysis (High-Res)				
Layout Detection	5.2	10.5	21.2	$O(P)$
OCR (Dense Text)	24.5	48.2	98.4	$O(P \times T)$
Visual Embedding	12.5	25.8	52.6	$O(N_{\text{img}})$
Structure & Graph Construction				
DHP Tree Parse	0.5	1.0	2.1	$O(N_{\text{blk}})$
Graph Build	0.2	0.4	0.9	$O(N + E)$
Total Indexing Time	42.8	85.8	175.2	Linear w.r.t Pages

Table 13: Offline indexing runtime and scalability as a function of document length. We report stage-wise latency (layout detection, OCR, visual embeddings, DHP parsing, and graph building) for 5/10/20-page documents and provide complexity terms indicating the main cost drivers.

I Runtime and Scalability Analysis

We analyze the computational cost of **HiKEY** under a realistic industrial setting. Measurements are conducted on a single NVIDIA H100 (80GB) GPU with 144 DPI rendering and high-precision OCR settings; peak GPU memory during offline indexing is approximately 27 GB.

I.1 Offline Indexing Cost

Table 13 breaks down the end-to-end processing runtime by document length, including high-resolution layout detection, OCR, visual embedding, DHP tree parsing, and graph construction. The full offline pipeline amortizes to roughly 8–9 s/page on H100 in this high-precision configuration; the core DHP tree parsing itself is lightweight (about 0.1 s/page), so the bulk of the cost is paid once by OCR and visual embedding at index time, not at query time.

Findings and Cost Justification. Over 90% of the total indexing time is allocated to visual analysis (layout, OCR, embeddings). This reflects the unavoidable cost of preserving complex industrial document structures in high resolution. Crucially, the core logic of **HiKEY**—tree parsing and graph construction—is extremely lightweight ($< 2\%$ of total time) and does not create a bottleneck. This confirms that **HiKEY** efficiently organizes expensive vision-derived signals into a structured index without adding significant overhead.