Towards Agentic Document-Grounded Information Seeking

Anonymous ACL submission

Abstract

Document Question Answering (DocQA) focuses on answering questions grounded in given documents, yet existing DocQA agents lack effective tool utilization and largely rely on closed-source models. In this work, we introduce **DocDancer**, an end-to-end trained open-source Doc agent. We formulate DocQA as an information-seeking problem and propose a tool-driven agent framework that explicitly models document exploration and comprehension. To enable end-to-end training of such agents, we introduce an *Exploration-then-Synthesis* data synthesis pipeline that addresses the scarcity of high-quality training data for DocQA. Training on the synthesized data, the trained models on two long-context document understanding benchmarks, MMLongBench-Doc and DocBench, show their effectiveness. Further analysis provides valuable insights for the agentic tool design and synthetic data.[1]

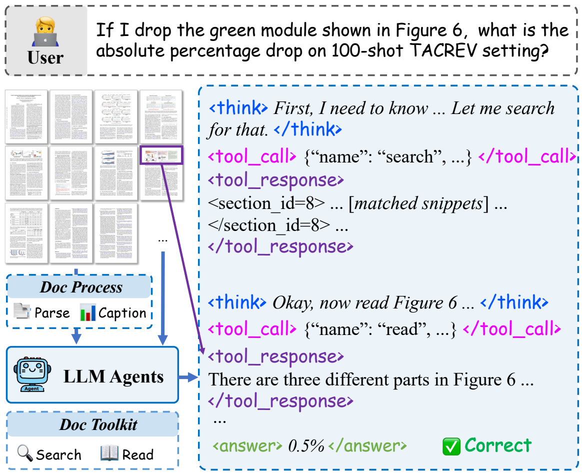

Figure 1: The overall of **DocDancer** for document-grounded information seeking, where *search* and *read* tools for effective document retrieval and comprehension over processed documents.

1 Introduction

Understanding and answering questions over long, multi-modal documents is a critical capability for real-world intelligent systems (Tkaczyk et al., 2015; Liu et al., 2025b). Document Question Answering (DocQA) lies at the core of document-centric intelligence, enabling models to access, reason over, and synthesize information from complex and heterogeneous document sources.

Existing DocQA methods can be broadly categorized into three paradigms. The first paradigm relies on optical character recognition (OCR) to convert documents into plain text, which is then processed by downstream language models (Xu et al., 2020). The second paradigm adopts embedding-based retrieval mechanisms, most commonly instantiated through retrieval-augmented generation (RAG), to identify and incorporate relevant document segments during inference (Saad-Falcon et al., 2024). More recently, agent-based paradigms have gained increasing attention, as they better support complex scenarios that require iterative exploration, tool invocation, and multi-step reasoning over long and structured documents (Sun et al., 2025a; Zhu et al., 2025). Recent advances in large language models (LLMs) (Team, 2025; Liu et al., 2025a) enable such agents to dynamically decompose queries, interact with documents, and adapt to intermediate observations, alleviating the limitations of OCR- and RAG-based approaches. Despite their promise, existing DocQA agents are typically implemented as prompt-based pipelines, with limited learning of autonomous agentic behaviors.

In contrast, we aim to train the **first** end-to-end DocQA agent model that is explicitly grounded in information-seeking principles, moving beyond prompt-based agent designs. We first formulate DocQA as an agentic information-seeking problem and design a tool-centric agent framework that decomposes document understanding into two complementary capabilities. Specifically, we introduce

[1] Our code, data, and trained models will be released upon acceptance.

efficient `search` tools for global information acquisition and fine-grained `read` tools for localized comprehension. This design enables the agent to actively explore long documents, iteratively refine its hypotheses, and dynamically adapt its strategy based on intermediate observations. Notably, when instantiated with a proprietary LLM, our framework achieves state-of-the-art performance and exceeds reported human-level performance.

Furthermore, a key bottleneck in training such agent models is the scarcity of high-quality DocQA pairs (Huang et al., 2025), as most publicly available datasets provide only test splits and lack sufficiently annotated training data. To address this challenge, we propose an *Exploration-then-Synthesis* DocQA generation pipeline that progressively enhances QA pairs from easy to hard. Specifically, we first explore a source document through intent-guided, tool-augmented interactions to collect grounded evidence (the *Exploration* stage), and then synthesizes high-quality document-grounded QA pairs via multi-observation reasoning (the *Synthesis* stage). We then train our DocQA agent, **DocDancer**, on the synthesized dataset, instantiating it with two open-source backbones, Qwen3-4B-Thinking-2507 and Qwen3-30B-A3B-Thinking-2507 (Team, 2025). Despite being trained with **only 5,000** instances, both variants achieve competitive performance, with the 30B-A3B model attaining state-of-the-art results in several settings.

Extensive experiments are conducted on two long-context document understanding benchmarks, MMLongBench-Doc (Ma et al., 2024) and DocBench (Zou et al., 2025). The results demonstrate the effectiveness of the proposed **DocDancer**. Further analyses provide insights into document parsing strategies, tool design, and the role of synthetic data in agent learning. In summary, our contributions are three-fold:

- **Effective Agentic DocQA Framework**: We propose a tool-driven DocQA agent framework grounded in information-seeking principles, which achieves SOTA performance when paired with a proprietary LLM.
- **Autonomous Data Synthesis Pipeline**: We introduce an *Exploration-then-Refine* data synthesis pipeline that generates high-quality training data for learning agentic behaviors.
- **Empirical Performance**: Our method achieves state-of-the-art results and provides practical insights into effective and efficient agentic system design.

2 Related Work

Document Question Answering Methods. Traditional DocQA methods rely on OCR-based pipelines (Ding et al., 2022) or end-to-end vision–language models (Sukh, 2025; Hu et al., 2025), but both are constrained by limited input length and struggle with long documents (Ma et al., 2024; Zou et al., 2025; Dong et al., 2025a). Retrieval-augmented generation (Zhang et al., 2024; Dong et al., 2025a,b) improves scalability, yet most approaches decouple retrieval and reasoning in a single-shot manner, making them brittle to retrieval errors and ineffective for complex, multi-step queries (Zhang et al., 2025). Recent agent-based DocQA systems (Wu et al., 2025c; Sun et al., 2025a; Dong et al., 2025c) address these issues through iterative document navigation and reading, but they predominantly depend on prompt-engineered, closed-source LLMs. In this work, we aim to train an open-source document agent with learnable behaviors for robust and scalable DocQA.

Synthetic Data for Agent Training. High-quality training data is critical for training agents. Due to its scalability, rapid iteration, and inherent trainability, synthetic data offers significant advantages over manually annotated data, serving as a highly effective alternative to human-labeled datasets for agent learning (Liu et al., 2025a; Team et al., 2025b). Prior work has demonstrated that large-scale agent-synthesized data can be effectively generated for search agents (Wu et al., 2025a; Li et al., 2025b; Tao et al., 2025), code agents (Yang et al., 2025), GUI agents (Sun et al., 2025b; Guo et al., 2025a) and general-purpose agents (Fang et al., 2025; Prabhakar et al., 2025). In contrast, this work focuses on the DocQA agent setting. Existing DocQA datasets are primarily constructed through semi-automated (Van Landeghem et al., 2023; Dong et al., 2025b) or expert-annotated (Hendrycks et al., 2021; Deng et al., 2025) processes, both of which require substantial human involvement or result in questions that lack sufficient depth. Inspired by advances in search agents, we formulate DocQA as an agentic information-seeking problem, with the goal of synthesizing high-quality training data tailored for DocQA agents.

3 Methods

3.1 Agent Setup

Framework. We adopt the vanilla ReAct (Yao et al., 2022) as the agent's framework, which synergizes reasoning and acting. In this paradigm, the agent generates both a reasoning trace (thought), τ, and a subsequent action, a, in an interleaved manner. This process forms a trajectory, $\mathcal{H}_T$, which is a sequence of thought-action-observation triplets:

$$\mathcal{H}_T = (\tau_0, a_0, o_0, \ldots, \tau_i, a_i, o_i, \ldots, \tau_T, a_T), \quad (1)$$

where a_T represents the final answer to the given task. At any given step $t \leq T$, the agent's policy, π, generates the current thought τ_t and action a_t based on the history of all previous interactions, $\mathcal{H}_{t-1}$:

$$\tau_t, a_t \sim \pi(\cdot|\mathcal{H}_{t-1}). \quad (2)$$

Inspired by *The Bitter Lesson* (Sutton, 2019), we employ a single-agent setup with carefully selected, highly effective tools, rather than relying on multi-agent designs or test-time scaling.

Document Processing. Prior works (Sun et al., 2025a) show that an XML-based hierarchical representation for document outlines that organizes parsed content into nested trees, using sections as partitioning units and elements such as text, images, and tables as nodes. While this structure enables efficient positioning and search, it suffers from structural and content inaccuracies and does not incorporate retrieval-aware visual information, which limits its applicability to agent-based processing of long, visually rich documents. To address these issues, we substantially enhance the document outline. For content accuracy, we leverage MinerU2.5 (Niu et al., 2025) for high-precision layout analysis and extraction, defining 17 element types and enriching outline nodes with layout and semantic attributes while removing structurally irrelevant elements such as headers and footers. For structural accuracy, title elements are visually cropped and clustered to infer hierarchical levels, enabling fine-grained section segmentation and reducing information loss in long documents. To improve visual retrieval, we generate captions for images and charts using an multimodal model M_m and incorporate them as auxiliary information, allowing the outline to better align and retrieve visual content.

Tool Design. We point out that DocQA can be naturally formulated as an *agentic information-seeking* task in which the external information source is restricted to the given documents. Accordingly, our tool design aims to enable agents to efficiently and effectively locate and extract relevant information from documents, while keeping the overall toolkit complexity low to ensure ease of use for agent models. Specifically, we design the following two tools for DocDancer:

- ***Search.*** Conducts keyword-based full-text search over the given documents, returning the section IDs, page numbers, and surrounding text snippets for each match. A visible window is used to constrain the snippet length for efficient localization. This tool provides the agent with global textual signals for guiding subsequent information access.
- ***Read.*** Given a goal and a set of section IDs, the tool performs fine-grained reading to extract goal-relevant information from the specified sections. This includes (i) local textual information, consisting of all text within the section; (ii) local visual information, consisting of images and tables within the section, together with a page-level screenshot that captures the full layout of the page containing the section. Subsequently, a multimodal summarization model M_m is used as an auxiliary reader to jointly integrate textual and visual inputs and return consolidated goal-relevant content.

This design deliberately integrates textual and visual signals, capturing both localized evidence and global layout cues, while keeping the toolkit limited to two tools to facilitate efficient utilization.

3.2 Data Synthesis

It is crucial to curate complex and diverse Document DocQA pairs that are capable of eliciting multi-step reasoning, goal decomposition, and rich interaction trajectories. To this end, we first construct a broad and heterogeneous collection of PDF documents to serve as the grounding corpus for question answering. We then synthesize QA pairs based on these documents, ensuring coverage of diverse reasoning patterns and document structures.

Sources. To construct a robust and diverse dataset for document-based question answering, we select four representative datasets, LongDocURL (Deng et al., 2025), MMDocRAG (Dong et al., 2025b), CUAD (Hendrycks et al., 2021) and DUDE (Van Landeghem et al., 2023), that cover long-context understanding, multimodal retrieval, legal expertise, and complex layout analysis. These sources provide the foundational PDF documents used for our automated QA generation pipeline. The distribution of the collected PDF documents is illustrated in Figure 3.

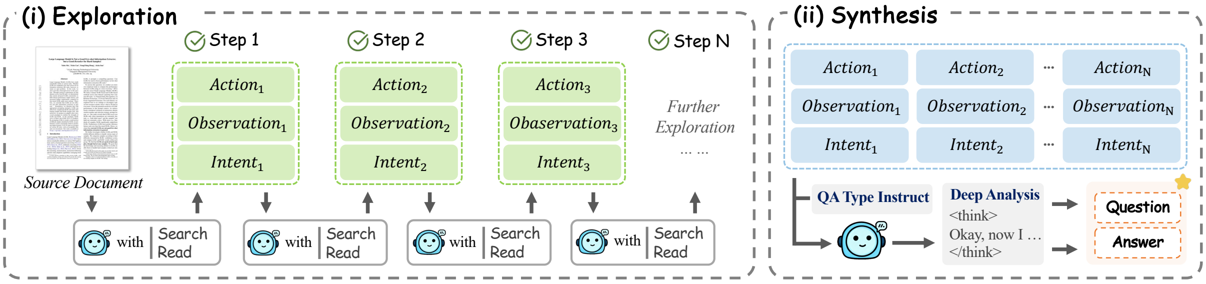

Figure 2: **Overall of the *Exploration-then-Synthesis* framework**. (i) *Exploration* stage iteratively interacts with the source document through Action(u)–Observation(y)–Intent(i) steps. (ii) *Synthesis* stage aggregates the collected evidence to generate the final question and answer. We present a concrete case illustrating the whole generation process in Appendix A.

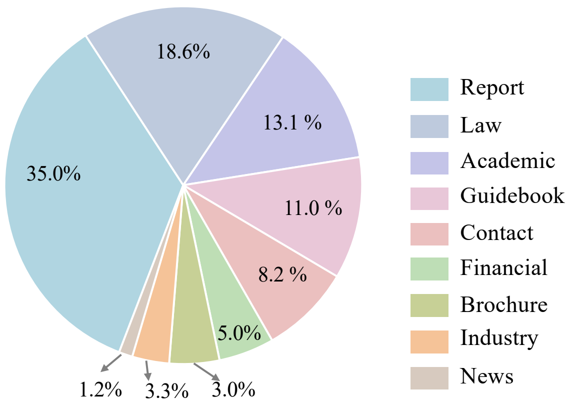

Figure 3: Distribution of document used to synthesise.

Exploration-then-Synthesis Framework. We propose a two-stage framework for DocQA generation, consisting of an *Exploration* Stage and a *Synthesis* Stage as shown in Figure 2. The overall objective is to transform a source document into a diverse and high-quality set of grounded QA pairs through iterative interaction and reasoning.

***Exploration* Stage.** Given a source document $\mathcal{D}$, utilze an LLM M_e to iteratively interact with $\mathcal{D}$ and collect information relevant to potential QA pairs. Conditioned on the interaction history h_t and the document $\mathcal{D}$, we employ model M_s jointly generates an intent-action pair (i_t, a_t):

$$(i_t, u_t) \sim \pi_{M_e}(i, u \mid h_t, \mathcal{D}), \tag{3}$$

where i_t denotes the exploration intent and $u_t \in \mathcal{A}$ corresponds to invoking a document-grounded tool such as *Search* or *Read*, which is the same as the agent's tool action. The construction of a question implicitly induces the strategy required to resolve it. The explicit modeling of intent helps prevent uninformative exploration, guiding the agent toward more concrete, goal-directed trajectories (Pahuja et al., 2025). Executing action a_t yields an observation:

$$y_t = \mathcal{T}(a_t, \mathcal{D}), \tag{4}$$

where $\mathcal{T}$ denotes the document interaction interface. The interaction history is then updated as:

$$h_{t+1} = h_t \cup \{(i_t, u_t, y_t)\}, \tag{5}$$

and the intent i_{t+1} may be revised based on the newly acquired information.

This process is repeated for multiple steps, enabling the agent to progressively refine its understanding of the document and uncover diverse and informative content. The explicit modeling of intent allows for flexible and open-ended exploration, permitting additional interactions when necessary.

The output of the exploration stage is a trajectory

$$\xi = \{(i_t, u_t, y_t)\}_{t=1}^{T}, \tag{6}$$

which serves as structured evidence for downstream QA generation.

In the exploration stage, each exploration step can be viewed as a random walk over the knowledge graph implicitly embedded in the entire document. When the number of such walks is sufficiently large, this process can, in principle, reconstruct the underlying document-level knowledge graph in a reverse manner. This idea is conceptually aligned with prior work on QA generation based on knowledge graphs in web search agent (Li et al., 2025b,a). We do not explicitly construct a document-level knowledge graph in advance, as such an approach would incur substantial engineering complexity and overhead. Instead, our method adopts a more lightweight design that is nevertheless capable of generating challenging DocQA pairs, achieving a better trade-off between efficiency and effectiveness.

***Synthesis* Stage.** Given the exploration trajectory ξ, the agent enters the synthesis stage to generate document-grounded QA pairs. A synthesis model M_s performs reasoning over the accumulated observations and generates a QA pair:

$$(q, a) \sim M_s(\xi, \mathcal{D}), \tag{7}$$

This stage emphasizes *(i)* reasoning over multiple observations collected during exploration, *(ii)* grounding both questions and answers in the source document, and *(iii)* producing semantically coherent and well-formed outputs. The final output is a set of K, document-grounded QA pairs:

$$\mathcal{QA} = \{(q_k, a_k)\}_{k=1}^{K}, \tag{8}$$

which can be used for training an agent. We employ a strong open-source model M_t to perform rejection sampling over these QA pairs, $\mathcal{QA}$, thereby obtaining high-quality training trajectories.

3.3 Agent Training

Following the empirical findings of (Chen et al., 2023), twe mask loss contributions from observation tokens to mitigate interference from external feedback during training, which has been shown to improve both performance and robustness. Given the task context $\mathbf{tc}$ and the complete execution trajectory $\mathcal{H} = (x_0, ..., x_{n-1}, x_n)$, where each $x_i \in \{\tau, \alpha, o\}$, the loss L is computed as follows:

$$L = -\frac{1}{\sum_{i=1}^{|\mathcal{H}|} \mathbb{I}[x_i \neq o]} \sum_{i=1}^{|\mathcal{H}|} \mathbb{I}[x_i \neq o] \cdot \log \pi_\theta(x_i \mid \mathbf{tc}, x_{<i}) \tag{9}$$

Here, $\mathbb{I}[x_i \neq o]$ filters out tokens corresponding to external feedback, ensuring the loss is computed only over the agent's decision steps.

4 Experiments

In this section, we aim to answer the following research questions (**RQs**):

- **RQ1**: How effective is the proposed information-seeking agent framework for DocQA?
- **RQ2**: How effective is the proposed synthetic data pipeline for training open-source DocQA agents?
- **RQ3**: Which components of the agent framework contribute most to performance?
- **RQ4**: How does the proposed DocDancer in qualitative evaluations?

4.1 Experimental Setup

We fine-tune Qwen3-30B-A3B-Thinking-2507 and Qwen3-4B-Thinking-2507 on our dataset, resulting in DocDancer. Our detailed implementation is provided in Appendix B, trained with only 5,000 agent trajectories.

Benchmarks. We evaluate the proposed DocAgent on two multimodal long-context document question answering benchmarks: MMLongBench-Doc (Ma et al., 2024) and DocBench (Zou et al., 2025). MMLongBenchDoc comprises 135 documents with an average length of 47.5 pages, featuring rich layouts and multimodal components across seven diverse domains. The dataset includes 1,091 questions derived from multiple sources, such as text, tables, charts, and images, with 33% involving cross-page reasoning. DocBench consists of 229 real-world documents and 1,082 questions, covering five domains and four major question types.

Metrics. For MMLongBench-doc, we follow the official evaluation protocol. Answers are extracted using `GPT-4.1` and evaluated with rule-based scoring to compute F1 (F_1) and Accuracy (*acc*). To mitigate extraction errors and improve robustness to diverse response formats, we additionally employ an LLM-as-Judge (*LasJ*) setting, where `gpt-4o` assigns binary scores using carefully designed prompts. For DocBench, we likewise adhere to the official evaluation procedure, using the provided instructions to guide `GPT-4.1` for assessment.

Baselines. We compare our approach with the following three categories of baselines: (1) VLM-based methods: Following the setting of MMLongBench-Doc, PDF pages are scanned at 144 DPI and used as input to the VLM. (2) OCR-based methods: Text is extracted from documents using an OCR tool, and the parsed plain text is provided to a LLM for answering. Text beyond the model's context length is truncated. (3) RAG-based methods: In this category, we compare existing RAG frameworks for DocQA, including VisRAG (Yu et al., 2024), Colpali (Faysse et al., 2024), M3DocRAG (Cho et al., 2025), MMGR (Wan and Yu, 2025), and RAGAnything (Guo et al., 2025b). (4) Agent-based methods: We include several recent and well-performing training-free agentic frameworks, namely Doc-React (Wu et al., 2025c), MDocAgent (Han et al., 2025), MACT (Yu et al., 2025), SimpleDoc (Jain et al., 2025), DocLens (Zhu et al., 2025), and DocAgent (Sun et al., 2025a). The detailed introduction of the baseline is

Method	Model	MMLongBench-Doc *acc*	MMLongBench-Doc F_1	MMLongBench-Doc *LasJ*	DocBench *LasJ*
VLM Baseline					
Naive VL (Ma et al., 2024)	`GPT-4o`	42.8	44.9	–	63.1
Naive VL (Zhu et al., 2025)	`Gemini-2.5-Pro`	–	–	58.1	–
OCR-based Baseline					
fitz[2]	`GPT-4`	–	–	–	67.9
Tesseract (Smith, 2007)	`GPT-4o`	30.1	30.5	–	–
Tesseract (Smith, 2007)	`Gemini-2.0-Flash`	39.6	37.2	–	–
RAG-based Baseline					
VisRAG (Yu et al., 2024)	`GPT-4o`	29.0	27.8	–	–
Colpali (Faysse et al., 2024)	`GPT-4o`	32.2	30.8	–	–
M3DocRAG w/ ColPali (Cho et al., 2025)	`Qwen2-VL-7B`	31.4	36.5	–	–
RAGAnything (Guo et al., 2025b)	`GPT-4o-mini`	42.8	–	–	63.4
Prompt-based Agent					
Doc-React (Wu et al., 2025c)	`GPT-4o`	38.1	38.3	–	–
MDocAgent (Han et al., 2025)	`GPT-4o`	42.0	–	–	–
MACT (Yu et al., 2025)	`MiMo-VL-7B`	47.4	–	–	–
SimpleDoc (Jain et al., 2025)	`Claude-4-Sonnet`	–	–	58.6	–
SimpleDoc (Jain et al., 2025)	`Gemini-2.5-Pro`	–	–	56.6	–
DocLens (Zhu et al., 2025)	`Claude-4-Sonnet`	–	–	63.3	–
DocLens (Zhu et al., 2025)	`Gemini-2.5-Pro`	–	–	**67.6**	–
DocAgent (Sun et al., 2025a)	`GPT-4o`	51.8	49.1	–	79.9
DocAgent (Sun et al., 2025a)	`Claude-3.5-Sonnet`	**57.3**	54.1	–	–
Ours					
DocDancer	`GPT-4o`	52.3	50.8	59.2	73.5
	`Gemini-2.5-Pro`	56.3	55.3	65.9	79.9
	`GPT-5.2`	57.0	**56.8**	**67.6**	**85.5**
	`Qwen3-4B (ft)`	48.4	49.2	59.4	79.8
	`Qwen3-30B-A3B (ft)`	54.4	53.9	65.3	81.2
Human Baseline	–	65.8	66.0	–	81.2

Table 1: **Performance comparison** across two long-context understanding benchmarks. The best results among all methods are **bolded** and the second-best results are underlined.

provided in Appendix C.

4.2 Overall Performance (RQ1)

We evaluate our agent framework against OCR-based, RAG-based, and prompt-based baselines on long-document DocQA benchmarks. Based on the experimental results in Table 1, we draw the following observations. **First**, agent-based approaches substantially outperform VLM-based methods, OCR-based baselines, and RAG-based baselines across evaluated benchmarks, highlighting the advantage of explicit tool use and iterative reasoning for long-context document understanding. **Second**, under the same backbone, our single-agent framework matches or surpasses multi-agent systems. In particular, on MMLongBench-Doc, DocDancer with GPT-5.2 attains 56.8 F_1 / 67.6 *LasJ*, outperforming all prior methods, and on DocBench, it reaches 85.5, exceeding the human baseline by 4 points. **Third**, models trained on our synthetic DocQA dataset demonstrate strong generalization and data efficiency. Even with relatively small model sizes, such as 30B-A3B and 4B, the resulting agents achieve performance competitive with closed-source models. These results indicate that training agentic capabilities on smaller-scale models is both feasible and highly valuable, substantially lowering the barrier to building effective document-understanding agents.

4.3 Effectiveness of Synthetic Data (RQ2)

Overall Performance. We investigate whether the *Exploration-then-Synthesis* data generation pipeline provides effective supervision for learning agentic behaviors, and whether models trained solely on the synthesized data achieve strong performance compared to existing open-source QA pairs. In Figure 5, we use the same PDF sources (Section §3.2) and construct two training sets of **equal size** (5,000 instances): one from our synthesized QA

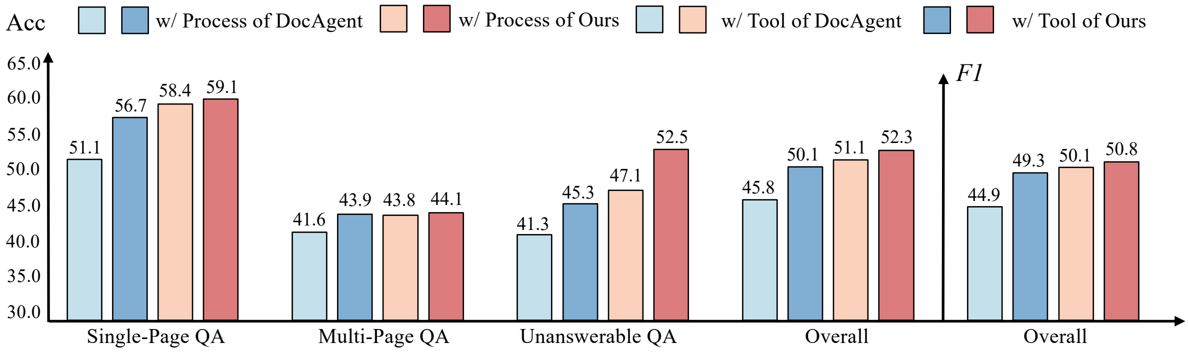

Figure 4: **Ablation study** on document parsing and tools.

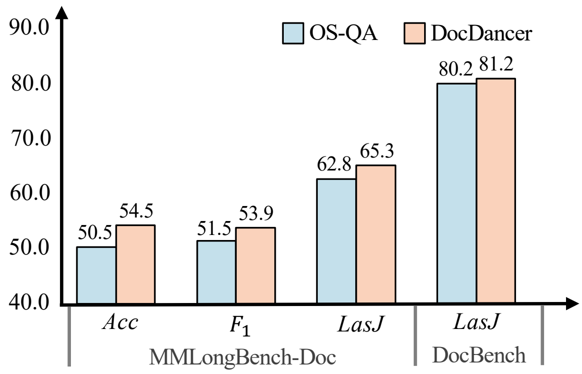

Figure 5: Performance comparison between models trained on **our synthesized QA data** and **open-source QA data.**

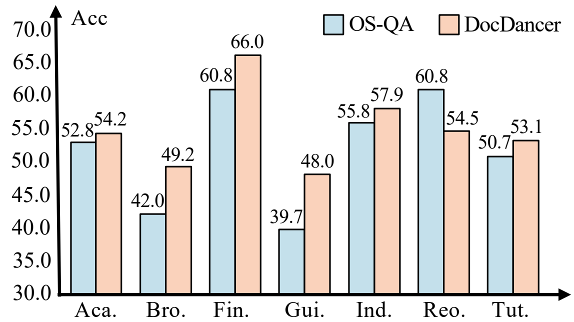

Figure 6: **Detailed domain-wise performance** comparison on MMLongBench-Doc between DocDancer and the model trained on OS-QA.

data and the other from human-annotated QA data provided with the PDFs (**OS-QA**). Both models are trained on Qwen3-30B-A3B-Thinking-2507. Overall, DocDancer consistently outperforms OS-QA across all metrics and benchmarks, demonstrating the effectiveness of our data synthesis strategy.

Detailed Results on Domains. Figure 6 reports domain-level results on MMLongBench-Doc. DocDancer consistently outperforms the QA baseline across all document domains, including Academic, Financial, Industry, and Report. The gains are more pronounced in structurally complex domains that require iterative information seeking and fine-grained reasoning. Overall, the results indicate that DocDancer generalizes well across diverse document types and is robust to domain variation.

4.4 Influence of Agentic Tools (RQ3)

We conduct ablation studies on document processing for outline construction and tool usage in Figure 4. The baseline is the Actor Agent from DocAgent (Sun et al., 2025a). For outline construction, DocAgent relies on Adobe PDF Extract as well as DocXChain (Yao, 2023) and PyMuPDF. In contrast, our enhanced method employs MinerU2.5 (Niu et al., 2025) for outline generation. The results demonstrate that, when combined with the same tools, our processing approach consistently outperforms the baseline, confirming that MinerU2.5 produces higher-quality document outlines. Regarding tool usage, DocAgent utilizes five tools: *search*, *get_section_content*, *get_image*, *get_page_images*, and *get_table_image*. In comparison, we only use two tools, *Search* and *Read*, following the principle of simplicity. Despite this reduced tool set, our approach achieves better performance when combined with either our own outline or the outline generated by DocAgent. The best results are obtained by combining our outline construction with our tool design, demonstrating their complementary effects. Furthermore, we conduct an ablation study on the external model used by the *Read* tool. Our default configuration, M_m employs Qwen3-VL-235B-A22B-Instruct. Replacing it with Gemini-3-Pro yields a modest overall improvement of 0.2 accuracy points on DocBench (Figure 8), with gains in Government, Law, and News domains. These results indicate that our tool design is robust and does not depend on an excep-

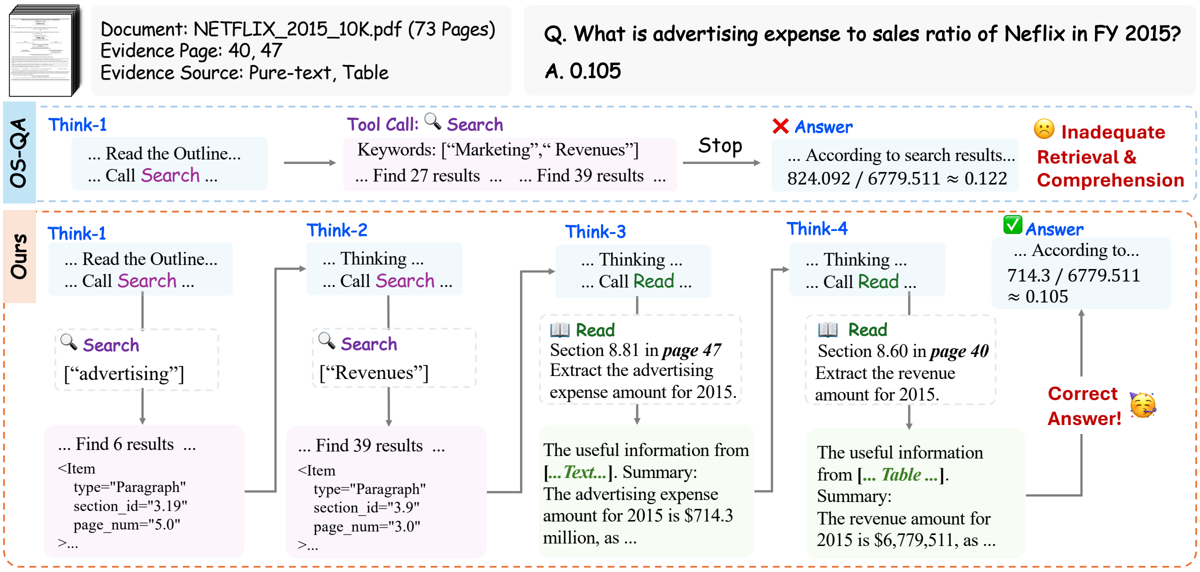

Figure 7: **A case study** demonstrating that our proposed DocDancer successfully performs multi-round information gathering to reach the correct answer, as illustrated in Table 3 in detail, whereas OS-QA produces an incorrect result.

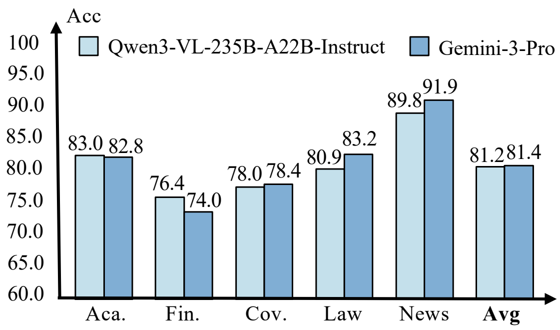

Figure 8: Results on DocBench across various domains **using different models used by *Read* tool**. We report the generalized accuracy of five types of document domains, including Academia (Aca.), Finance (Fin.), Government (Gov), Law, and News.

tionally strong external model.

4.5 Qualitative Analysis (RQ4)

We present a case study of a financial task on a 73-page document from MMLongBench-Doc, as illustrated in Figure 7. Answering this question requires locating advertising expense and revenue figures from different sections of the document and performing a numerical computation. The baseline model, which is trained on OS-QA relies on keyword-based retrieval and retrieves passages related to “marketing” and “revenues”. Due to insufficient grounding, it incorrectly uses a marketing expense figure as a proxy for advertising expense, yielding an erroneous ratio of 0.122. This failure illustrates the limitation of single-pass retrieval and shallow aggregation when fine-grained financial concepts are required. In contrast, DocDancer performs multi-round, question-driven information gathering. It first retrieves and reads the section explicitly reporting advertising expense for FY 2015 ($714.3M), and then independently extracts the total revenue from a separate tabular section ($6,779.5M). By grounding each value to its corresponding evidence and verifying semantic relevance, the system computes the correct ratio of $714.3/6{,}779.5 \approx 0.105$. It demonstrates that accurate document-level financial question answering benefits from our synthetic data, which enables the construction of **domain-specific expert-level** supervision beyond ordinary human annotations.

5 Conclusion

We propose DocDancer, an end-to-end trained agentic model for document question answering that formulates DocQA as an information-seeking process. By introducing a tool-centric framework with complementary search and read operations, DocDancer enables effective exploration and comprehension of long, structured documents. To mitigate the lack of high-quality supervision, we further design an Exploration-then-Synthesis data pipeline that generates compact yet effective training data for learning agentic behaviors. Experiments on MMLongBench-Doc and DocBench demonstrate that DocDancer achieves strong and competitive performance, validating the effectiveness of agentic information-seeking for document understanding.

Limitations

This work still has several limitations. First, our experiments are conducted only on Qwen3-30B-A3B-Thinking-2507 and Qwen3-4B-Thinking-2507; we do not evaluate the proposed method on larger-scale models or models from other families. Second, we focus exclusively on supervised fine-tuning (SFT) and do not explore agentic reinforcement learning (RL). Third, we do not further scale the training data, and thus do not investigate how the proposed method performs under larger or more diverse data.

Ethical Considerations

This work studies agentic document-grounded question answering using publicly available benchmarks and documents released for research purposes. The proposed *Exploration-then-Synthesis* pipeline generates synthetic question–answer pairs that are explicitly grounded in source documents and does not introduce new proprietary data or attempt to reproduce large portions of copyrighted text verbatim. While the method itself does not collect personal information, document-grounded agents may be applied to sensitive or private documents in downstream use; such applications require appropriate authorization and privacy safeguards. The synthesized data and trained models may inherit biases present in the underlying document sources, including domain and content imbalances. Finally, although improved document exploration capabilities could be misused if deployed irresponsibly, the strong grounding in retrieved evidence and our commitment to releasing code and data aim to support transparency, reproducibility, and responsible research use.

References

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. Fireact: Toward language agent fine-tuning. arXiv preprint arXiv:2310.05915.

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, pages 6178–6188.

Chao Deng, Jiale Yuan, Pi Bu, Peijie Wang, Zhong-Zhi Li, Jian Xu, Xiao-Hui Li, Yuan Gao, Jun Song, Bo Zheng, and 1 others. 2025. Longdocurl: a comprehensive multimodal long document benchmark integrating understanding, reasoning, and locating. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1135–1159.

Yihao Ding, Zhe Huang, Runlin Wang, YanHang Zhang, Xianru Chen, Yuzhong Ma, Hyunsuk Chung, and Soyeon Caren Han. 2022. V-doc: Visual questions answers with documents. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 21492–21498.

Kuicai Dong, Yujing Chang, Xin Deik Goh, Dexun Li, Ruiming Tang, and Yong Liu. 2025a. Mmdocir: Benchmarking multi-modal retrieval for long documents. arXiv preprint arXiv:2501.08828.

Kuicai Dong, Yujing Chang, Shijie Huang, Yasheng Wang, Ruiming Tang, and Yong Liu. 2025b. Benchmarking retrieval-augmented multimomal generation for document question answering. arXiv preprint arXiv:2505.16470.

Kuicai Dong, Shurui Huang, Fangda Ye, Wei Han, Zhi Zhang, Dexun Li, Wenjun Li, Qu Yang, Gang Wang, Yichao Wang, and 1 others. 2025c. Docresearcher: A unified system for multimodal document parsing and deep research. arXiv preprint arXiv:2510.21603.

Runnan Fang, Shihao Cai, Baixuan Li, Jialong Wu, Guangyu Li, Wenbiao Yin, Xinyu Wang, Xiaobin Wang, Liangcai Su, Zhen Zhang, and 1 others. 2025. Towards general agentic intelligence via environment scaling. arXiv preprint arXiv:2509.13311.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. 2024. Colpali: Efficient document retrieval with vision language models. arXiv preprint arXiv:2407.01449.

Xiangwu Guo, Difei Gao, and Mike Zheng Shou. 2025a. Auto-explorer: Automated data collection for gui agent. arXiv preprint arXiv:2511.06417.

Zirui Guo, Xubin Ren, Lingrui Xu, Jiahao Zhang, and Chao Huang. 2025b. Rag-anything: All-in-one rag framework. arXiv preprint arXiv:2510.12323.

Siwei Han, Peng Xia, Ruiyi Zhang, Tong Sun, Yun Li, Hongtu Zhu, and Huaxiu Yao. 2025. Mdocagent: A multi-modal multi-agent framework for document understanding. arXiv preprint arXiv:2503.13964.

D. Hendrycks, C. Burns, A. Chen, and S. Ball. 2021. Cuad: An expert-annotated nlp dataset for legal contract review. arXiv preprint arXiv:2103.06268.

Anwen Hu, Haiyang Xu, Liang Zhang, Jiabo Ye, Ming Yan, Ji Zhang, Qin Jin, Fei Huang, and Jingren Zhou. 2025. mplug-docowl2: High-resolution compressing for ocr-free multi-page document understanding. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5817–5834.

Tiancheng Huang, Ruisheng Cao, Yuxin Zhang, Zhangyi Kang, Zijian Wang, Chenrun Wang, Yijie Luo, Hang Zheng, Lirong Qian, Lu Chen, and 1 others. 2025. Airqa: A comprehensive qa dataset for ai research with instance-level evaluation. arXiv preprint arXiv:2509.16952.

Chelsi Jain, Yiran Wu, Yifan Zeng, Jiale Liu, Zhenwen Shao, Qingyun Wu, Huazheng Wang, and 1 others. 2025. Simpledoc: Multi-modal document understanding with dual-cue page retrieval and iterative refinement. arXiv preprint arXiv:2506.14035.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Jialong Wu, and 1 others. 2025a. Websailor-v2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. arXiv preprint arXiv:2509.13305.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, and 1 others. 2025b. Websailor: Navigating super-human reasoning for web agent. arXiv preprint arXiv:2507.02592.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. arXiv preprint arXiv:2512.02556.

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

Yubo Ma, Yuhang Zang, Liangyu Chen, Meiqi Chen, Yizhu Jiao, Xinze Li, Xinyuan Lu, Ziyu Liu, Yan Ma, Xiaoyi Dong, and 1 others. 2024. Mmlongbench-doc: Benchmarking long-context document understanding with visualizations. Advances in Neural Information Processing Systems, 37:95963–96010.

Junbo Niu, Zheng Liu, Zhuangcheng Gu, Bin Wang, Linke Ouyang, Zhiyuan Zhao, Tao Chu, Tianyao He, Fan Wu, Qintong Zhang, and 1 others. 2025. Mineru2. 5: A decoupled vision-language model for efficient high-resolution document parsing. arXiv preprint arXiv:2509.22186.

Vardaan Pahuja, Yadong Lu, Corby Rosset, Boyu Gou, Arindam Mitra, Spencer Whitehead, Yu Su, and Ahmed Hassan. 2025. Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents. In Findings of the Association for Computational Linguistics: ACL 2025, pages 6300–6323.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, and 1 others. 2025. Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. arXiv preprint arXiv:2504.03601.

Jon Saad-Falcon, Joe Barrow, Alexa Siu, Ani Nenkova, Seunghyun Yoon, Ryan A. Rossi, and Franck Dernoncourt. 2024. PDFTriage: Question answering over long, structured documents. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 153–169, Miami, Florida, US. Association for Computational Linguistics.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053.

Ray 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. IEEE.

Andriy Sukh. 2025. Ocr-free document understanding using vision-language models.

Li Sun, Liu He, Shuyue Jia, Yangfan He, and Chenyu You. 2025a. DocAgent: An agentic framework for multi-modal long-context document understanding. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 17712–17727, Suzhou, China. Association for Computational Linguistics.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, and 1 others. 2025b. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5555–5579.

Richard Sutton. 2019. The bitter lesson. Incomplete Ideas (blog), 13(1):38.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, and 1 others. 2025. Webshaper: Agentically data synthesizing via information-seeking formalization. arXiv preprint arXiv:2507.15061.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, and 1 others. 2025a. Kimi-vl technical report. arXiv preprint arXiv:2504.07491.

Qwen Team. 2025. Qwen3 technical report. Preprint, arXiv:2505.09388.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, and 1 others. 2025b. Tongyi deepresearch technical report. arXiv preprint arXiv:2510.24701.

Dominika Tkaczyk, Paweł Szostek, Mateusz Fedoryszak, Piotr Jan Dendek, and Łukasz Bolikowski. 2015. Cermine: automatic extraction of structured metadata from scientific literature. International Journal on Document Analysis and Recognition (IJDAR), 18(4):317–335.

Jordy Van Landeghem, Rubén Tito, Łukasz Borchmann, Michał Pietruszka, Pawel Joziak, Rafal Powalski, Dawid Jurkiewicz, Mickaël Coustaty, Bertrand Anckaert, Ernest Valveny, and 1 others. 2023. Document understanding dataset and evaluation (dude). In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 19528–19540.

Xueyao Wan and Hang Yu. 2025. Mmgraphrag: Bridging vision and language with interpretable multimodal knowledge graphs. arXiv preprint arXiv:2507.20804.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, and 1 others. 2025a. Webdancer: Towards autonomous information seeking agency. arXiv preprint arXiv:2505.22648.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025b. WebWalker: Benchmarking LLMs in web traversal. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10290–10305, Vienna, Austria. Association for Computational Linguistics.

Junda Wu, Yu Xia, Tong Yu, Xiang Chen, Sai Sree Harsha, Akash V Maharaj, Ruiyi Zhang, Victor Bursztyn, Sungchul Kim, Ryan A Rossi, and 1 others. 2025c. Doc-react: Multi-page heterogeneous document question-answering. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 67–78.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pretraining of text and layout for document image understanding. In Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, pages 1192–1200.

John Yang, Kilian Lieret, Carlos E Jimenez, Alexander Wettig, Kabir Khandpur, Yanzhe Zhang, Binyuan Hui, Ofir Press, Ludwig Schmidt, and Diyi Yang. 2025. Swe-smith: Scaling data for software engineering agents. arXiv preprint arXiv:2504.21798.

Cong Yao. 2023. Docxchain: A powerful open-source toolchain for document parsing and beyond. arXiv preprint arXiv:2310.12430.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In The eleventh international conference on learning representations.

Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, and 1 others. 2024. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. arXiv preprint arXiv:2410.10594.

Xinlei Yu, Chengming Xu, Zhangquan Chen, Yudong Zhang, Shilin Lu, Cheng Yang, Jiangning Zhang, Shuicheng Yan, and Xiaobin Hu. 2025. Visual document understanding and reasoning: A multi-agent collaboration framework with agent-wise adaptive test-time scaling. arXiv preprint arXiv:2508.03404.

Jinxu Zhang, Yongqi Yu, and Yu Zhang. 2024. Cream: coarse-to-fine retrieval and multi-modal efficient tuning for document vqa. In Proceedings of the 32nd ACM International Conference on Multimedia, pages 925–934.

Junyuan Zhang, Qintong Zhang, Bin Wang, Linke Ouyang, Zichen Wen, Ying Li, Ka-Ho Chow, Conghui He, and Wentao Zhang. 2025. Ocr hinders rag: Evaluating the cascading impact of ocr on retrieval-augmented generation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 17443–17453.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. Swift:a scalable lightweight infrastructure for fine-tuning. Preprint, arXiv:2408.05517.

Dawei Zhu, Rui Meng, Jiefeng Chen, Sujian Li, Tomas Pfister, and Jinsung Yoon. 2025. Doclens: A tool-augmented multi-agent framework for long visual document understanding. arXiv preprint arXiv:2511.11552.

Anni Zou, Wenhao Yu, Hongming Zhang, Kaixin Ma, Deng Cai, Zhuosheng Zhang, Hai Zhao, and Dong Yu. 2025. Docbench: A benchmark for evaluating llm-based document reading systems. In Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing, pages 359–373.

A Case Study of Synthetic Data

Figure 9 demonstrates how the *Exploration-then-Synthesis* framework iteratively navigates a 73-page document, aggregating heterogeneous evidence, text (in Sec. 2.43), charts (in Figure 1), and tables (in Table 1), scattered across disjoint pages (pp. 40, 41, 49) to synthesize a high-quality question that requires complex reasoning.

In the *Exploration* Stage, the agent generates a exploartion trajectory ξ via iterative (i_t, u_t) steps, effectively performing a "random walk" over the document's implicit knowledge graph. It aggregates heterogeneous evidence by bridging disjoint pages—linking visual trends in a chart (p. 40) with precise values in text (p. 49) and a table (p. 41). In the *Synthesis* Stage, the model M_s reasons over this accumulated trajectory to construct a complex multi-hop numerical question (Wu et al., 2025b). The final QA pair requires arithmetic calculation ($29.92\% - 15\% = 14.92\%$) rather than simple retrieval, ensuring deep document grounding and preventing shortcut learning.

B Implementation Details

B.1 Details on Prompts

The prompts for the DocDancer are shown in Figure 10.

B.2 Tool Schema

This section details the tool schemas provided to the agent. We designed two primary tools: `search` for keyword-based retrieval and `read` for extracting content from specific document sections. The specific JSON structures defining these functions are shown in Figure 11.

B.3 Training Details

We fine-tune Qwen3-30B-A3B-Think[3] and Qwen3-4B-Think[4] using the Megatron-LM framework (Zhao et al., 2024; Shoeybi et al., 2019). Both models are trained with a context length of 128k to support long-document processing tasks. We employ the AdamW optimizer with a precision-aware configuration and a cosine decay learning rate scheduler, featuring a peak learning rate of 1.0×10^{-5}, a minimum of 1.0×10^{-6}, and a 5% warmup phase. The global batch size is configured to 16 for the Qwen3-30B-A3B-Think and to 40 for Qwen3-4B-Think. For Qwen3-30B-A3B-Think, we apply an auxiliary loss coefficient of 10^{-3} to ensure balanced expert routing. We train both models for 10 epochs and selected the checkpoint with best performance.

[3]https://huggingface.co/Qwen/Qwen3-30B-A3B-Thinking-2507

[4]https://huggingface.co/Qwen/Qwen3-4B-Thinking-2507

B.4 Inference Details

vLLM framework (Kwon et al., 2023) is used for inference; we employ a temperature of 0.6, a top_p value of 0.95, and a presence penalty of 1.1.

B.5 Hyperparameter

By default, M_m is Qwen3-VL-235B-A22B-Instruct, and we analyze the effects of replacing it in Section 4.4. For M_t, we use the open-source and relatively strong model gpt-oss-120b to perform rejection sampling. Further analysis is provided in Table 2. First, our method substantially outperforms the base model without fine-tuning, demonstrating the effectiveness of the proposed training strategy. Second, our approach also surpasses the model trained with reject sampling, validating the quality of the synthesized question–answer data and showing that it can effectively elicit and enhance the model's performance. For M_s, we employ gpt-oss-120b in *Exploration-then-Synthesis* framework to synthesis data.

B.6 Details on Prompts for Data Synthesis

The prompts utilized for **Exploration** and **Synthetic** within the Exploration-then-Refine framework are presented in Figure 12 and Figure 13, respectively. Regarding the exploration configuration, we adjust the maximum exploration depth based on the complexity of the document sources. Specifically, we set the maximum sampling depth to 20 for LongDocURL and MMdocRAG, while for DUDE and CUAD, this limit is set to 15.

C Baselines

We compare DocDancer against a comprehensive set of baselines categorized into four groups:

Naive VLM Baselines. These methods evaluate the native long-context understanding capabilities of advanced VLMs. We directly feed PDF pages converted to images (144 DPI) into the models without external parsing or retrieval. Following the settings in MMLongBench-Doc (Ma et al., 2024),

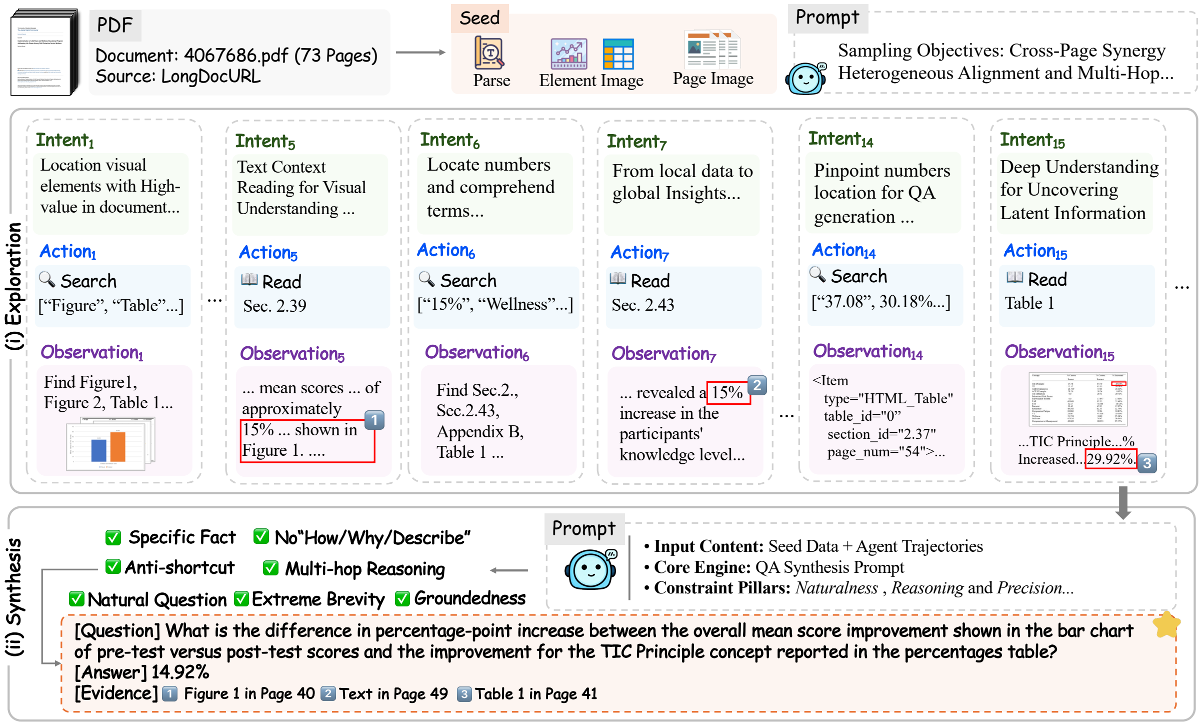

Figure 9: **A case study** of the Exploration-then-Synthesis framework generating a multi-hop numerical reasoning QA pair.

Method	Model	MMLongBench-Doc *acc*	MMLongBench-Doc F_1	MMLongBench-Doc *LasJ*	DocBench *LasJ*
DocDancer	`Qwen3-A3B-30B-Thinking`	39.2	36.4	46.9	74.1
DocDancer	`GPT-oss-120B`	52.3	53.0	59.8	80.8
DocDancer	`Qwen3-30B-A3B-Thinking (ft)`	54.4	53.9	65.3	81.2

Table 2: **Performance comparison** across two long-context understanding benchmarks.

Prompt

You are an expert research assistant tasked with answering questions based on document content.
You will be provided with an XML outline of the document. If you need more comprehensive, detailed, or accurate information from the document to fully address the user's query, you need to use the provided tool.

I've uploaded a document, and below is the outline in XML format: {document_outline}.
Answer the following question based on the content of the document: {question}.

Figure 10: System prompt for **DocDancer**.

we report *GPT-4o* [5] and *Gemini-2.5-Pro* [6].
OCR-based Baselines. These baselines treat the task as text-only QA by first extracting content using OCR engines. We pair *Tesseract* (Smith, 2007) and *PyMuPDF (fitz)* [7] with LLMs including *GPT-4*, *GPT-4o*, and *Gemini-2.0-Flash*.
RAG-based Baselines. We consider both visual and hybrid retrieval strategies:

- **Visual Retrieval: VisRAG** (Yu et al., 2024) and **ColPali** (Faysse et al., 2024) retrieve relevant page or patch-level visual evidence based on vision-centric embeddings, utilizing *GPT-4o* for response generation.

- **Hybrid Retrieval: M3DocRAG** (Cho et al., 2025) performs joint retrieval using a mul-

[5] https://platform.openai.com/docs/models/gpt-4o
[6] https://ai.google.dev/gemini-api/docs/models?#gemini-2.5-pro
[7] https://pymupdf.readthedocs.io/

timodal retriever with *Qwen2-VL-7B*. **RAGAnything** (Guo et al., 2025b) structures multimodal content as knowledge entities for cross-modal retrieval, using *GPT-4o-mini* as the backbone.

Prompt-based Agentic Baselines. We include state-of-the-art agent frameworks designed for document understanding:

- **Doc-React** (Wu et al., 2025c) employs an iterative decision-making process to balance information gain and uncertainty reduction (*GPT-4o*).
- **MDocAgent** (Han et al., 2025) utilizes a multi-agent system with five specialized roles for context retrieval (*GPT-4o*).
- **MACT** (Yu et al., 2025) introduces a multi-agent collaboration framework featuring adaptive test-time scaling (*MiMo-VL-7B* (Team et al., 2025a)).
- **SimpleDoc** (Jain et al., 2025) retrieves pages via *ColQwen2.5*, followed by LLM-based evidence selection (*Claude-4-Sonnet*, *Gemini-2.5-Pro*).
- **DocLens** (Zhu et al., 2025) operates as a tool-augmented multi-agent framework for focused reading (*Claude-4-Sonnet*, *Gemini-2.5-Pro*).
- **DocAgent** (Sun et al., 2025a) leverages a tree-structured document outline combined with retrieval tools (*GPT-4o*, *Claude-3.5-Sonnet*).

Tool Schemas

Search

```
{
    "type": "function",
    "function": {
        "name": "search",
        "description": "Find and extract all paragraphs and sections where any of the provided
             search terms appear",
        "parameters": {
            "type": "object",
            "properties": {
                "keywords": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    },
                    "description": "A list of query keywords for searching"
                }
            },
            "required": ["keywords"]
        }
    }
}
```

Read

```
{
    "type": "function",
    "function": {
        "name": "read",
        "description": "Read multiple sections by section IDs and extract useful information
             from all content contained in those sections, including both visual elements and
             textual elements.",
        "parameters": {
            "type": "object",
            "properties": {
                "section_ids": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    },
                    "description": "A list of section IDs to read from the document"
                },
                "goal": {
                    "type": "string",
                    "description": "The user goal that guides what useful information should be
                         extracted from the selected sections"
                }
            },
            "required": ["section_ids", "goal"]
        }
    }
}
```

Figure 11: Tool schema: *Search* and *Read*.

Exploration in Exploration-then-Refine Framework.

You are exploring a parsed PDF paper/report (outline + paragraphs + images + table snapshots + per-page screenshots). Your objective is to collect HIGH-QUALITY, GROUNDED evidence bundles that can later support HARD, multi-hop, visually grounded document Q&A synthesis.

Final QA Constraints You Must Enable (every eventual QA must satisfy ALL):
- Multi-page: Combining evidence from at least THREE different pages/sections, where the pieces of evidence are related.
- Multi-element: Contains at least two evidence source types (text paragraphs/charts/graphics/table screenshots and/or full-page layouts).
- Multi-hop: require at least TWO reasoning points (e.g. cross-reference + computation, footnote rule + chart reading, layout count + comparison, multiple related searches + readings).

Important: final questions should NOT rely on explicit document locations. Do NOT plan to use page numbers, section titles/IDs, or explicit figure/table numbers (e.g., "Figure <number>", "Table <number>") in the question. Instead, you must collect CONTENT-BASED CLUES that can uniquely identify the needed evidence:
- Caption keywords (short quote fragments), axis labels and units, legend item names, panel labels (a)/(b), distinctive row/column headers, and footnote phrases ("restated", "excluding", "unaudited", unit changes).

Exploration strategy using only search and read:
- Use search to find visuals, tables, footnotes, and their nearby discussion text. Start with keywords like: "Figure", "Fig.", "Chart", "Image", "Graph", "legend", "axis", "panel", "Table", "Note", "footnote", "restated", "excluding", "unaudited".
- For each promising hit, immediately read the covering section(s) with a goal that extracts:
  - The text content of the section in question.
  - Caption text, axis labels/units, legend items, and visual markers.
  - The exact table header path, target cell(s), and footnote rules.
  - The narrative claim/explanation that references the visual.
- Use the read function as much as possible, deliberately chain across pages.
- For conditional layout questions: identify a page by a unique visual cue, then use read to count visible tables/figures.

Avoid:
- Broad whole-document counts unless you turn them into comparative, multi-hop questions.
- Word-frequency counting.
- Repeating identical tool calls.
- Statistical analysis of the number of elements.

Every action during sampling should contribute to forming a future HARD, multi-page, multi-element, multi-hop document QA.

Figure 12: Prompt for *exploration* stage in Exploration-then-Refine framework.

Synthesis in Exploration-then-Refine Framework.

You must synthesize "document Q&A" training data based ONLY on the trajectory.

Hard Requirements (Strict):
- The output must be a JSON object containing only two fields: `question` and `answer` (no additional fields are allowed), and must be in English only.
- The question must be natural and unambiguous, containing only one question and corresponding to a single, unique answer.
- The question must not be a common-knowledge question; it must be impossible to answer based on the question alone and must be highly dependent on the document.
- Do not mention tools, sections, pages, section IDs, searching/reading actions, trajectories, or observations.
- The answer length should be limited to a single sentence, ideally a short phrase, entity, number, or list, and avoid simply using "yes/no" answers. The answer must be directly supported by evidence from the provided text and cannot be guessed randomly.

Mandatory Difficulty Constraints (every QA pair must satisfy all of the following):
1. **Multi-page:** The question requires evidence from at least two different pages/sections to answer, and the evidence must be logically related.
2. **Multiple Evidence Modalities:** The question must involve at least two types of evidence, such as text, charts, figures, tables, screenshots, and/or full-page layout cues, with a preference for covering visual elements.
3. **Multi-step Reasoning:** The question must require at least two reasoning steps (e.g., calculation + cross-validation, footnote rule application + chart reading, layout counting + comparison).

No Explicit Location References in the Question:
- Do not mention page numbers, section IDs, titles/IDs, or explicit figure/table numbers (e.g., "Figure <number>", "Table <number>").
- Instead, provide 1–3 content-based clues to help locate the evidence, such as: short title phrases, axis labels/units, legend item names, unique row names, footnote keywords, or distinctive layout hints (e.g., "the only multi-panel figure labeled (a) and (b)").
- When describing visual elements, do not directly copy long unique numbers or OCR-extracted long text strings from images (e.g., "an image showing the number 7,584,322,338"). Use specific entity names or semantic descriptions instead (e.g., "Apple's 2018 total sales table", "an image showing adjusted outstanding balances", or "the largest segment in the pie chart").

Preferred Question Templates (all templates must be cross-page + visual + multi-step):
- **Cross-page conditional layout:** Identify pages via unique visual cues and compare the number of visible objects across pages.
- **Textual claim + chart verification:** A narrative statement about a change/target that is verified using a chart and light calculation.
- **Table + chart consistency:** Compute a ratio/difference from a table and verify it against a data point in a chart on another page.
- **Footnote-constrained table + chart mapping:** Apply footnote/restatement/exclusion rules, then map the correct year/value to a chart on another page.
- **Table/Chart comprehension questions:** Locate tables and charts via text, then derive conclusions from table structure or chart visuals.
- **Unanswerable questions:** Questions that seem reasonable but are actually impossible to answer (e.g., questions about terms/entities that do not exist in the document). For these, the answer must be "Unanswerable".
- **Counting questions:** Count the occurrences of key local terms or entities in the document. Such questions should only be generated when there is sufficient and conclusive evidence.

Fallback Rule:
- If the current trajectory cannot support a question that satisfies all constraints, choose a different question.

After generating a question, perform a second-pass check and regenerate if the question falls into any of the following categories:
- Contains more than one question.
- Includes non-English languages or characters.
- Questions that can be answered based on an independent page/section.
- Common-sense questions unrelated to the document.
- Counting tasks spanning the entire document with a broad scope.
- Counting tasks involving Charts/Figures/Images/Tables.

For unanswerable questions, confirm that they are truly unanswerable. For counting questions, confirm completeness and answer accuracy. **Do not guess or fabricate answers under any circumstances.**

Figure 13: Prompts for Q&A *Synthesis* stage in Exploration-then-Refine framework.

Details of Case Study

Question

What is advertising expense to sales ratio of Netflix in FY 2015? Round your answer to three decimal places.

Answer

0.105

Step 1

<think>

We need to answer: “what is advertising expense to sales ratio of Netflix in FY 2015? Round your answer to three decimal places.” We need to locate advertising expense and sales (revenues) for FY 2015. From outline we have various sections. First, find advertising expense. The term “advertising” appears in sections. Search for “advertising”.

</think>

<tool_call>

`{ "arguments": "{ "keywords": ["advertising"] }", "name": "search" }`

</tool_call>

<tool_response>

A Document search for ‘advertising‘ found 6 results:

`<Item type="Paragraph" section_id="3.19" page_num="5.0">`... adjust pricing or service offerings ... `</Item>`

...

`<Item type="Paragraph" section_id="8.81" page_num="47.0">`

Marketing expenses consist primarily of advertising expenses... Advertising expenses were $714.3 million, $533.1 million and $404.0 million for the years ended December 31, 2015, 2014 and 2013, respectively.

`</Item>`

</tool_response>

Step 2

<think>

One of the results is in section_id “8.81” paragraph page_num “47.0”. It says: “Advertising expenses were $714.3 million, $533.1 million and $404.0 million for the years ended December 31, 2015, 2014 and 2013, respectively.” Thus advertising expense FY 2015 = $714.3 million.

Now sales (revenues) for FY 2015: In consolidated statements of operations (section 8.60) we have revenues $6,779,511 (in thousands). That’s $6,779,511 thousand = $6,779.511 million.

Alternatively, there may be a line item “Revenues” in Table 5 etc. Let’s locate “Revenues”.

</think>

<tool_call>

`{ "arguments": "{ "keywords": ["Revenues"] }", "name": "search" }`

</tool_call>

<tool_response>

A Document search for ‘Revenues‘ found 39 results:

`<Item type="Paragraph" section_id="3.9" page_num="3.0">`

...

`<Item type="HTML_Table" table_id="5" section_id="8.20" page_num="19.0">` ...

Revenues (2015): $ 6,779,511 ...

`</Item>`

...

</tool_response>

Continued on next page...

Continued from previous page

Details of Case Study (Continued)

Step 3

<think>
We have advertising expense: $714.3 million (2015). We need to compute ratio = advertising expense / revenue. Revenue for 2015: $6,779,511 thousand = $6,779.511 million. Thus ratio $= 714.3/6,779.511 =?$ ... First, we should cite sources: advertising expense paragraph (8.81) and revenue table.
Let's extract the advertising expense paragraph text to be sure.
</think>
<tool_call>
```
{ "arguments": "{ "section_ids": ["8.81"], "goal": "Extract the advertising expense amount for 2015" }", "name": "read" }
```
</tool_call>
<tool_response>
The useful information from the document section (section_id=8.81) for user goal ‘Extract the advertising expense amount for 2015‘ is as follows:
Evidence in document:
Marketing expenses consist primarily of advertising expenses... Advertising expenses were $714.3 million, $533.1 million and $404.0 million for the years ended December 31, 2015, 2014 and 2013, respectively.
Summary:
The advertising expense amount for 2015 is $714.3 million...
</tool_response>

Step 4

<think>
I have successfully extracted and verified the advertising expense ($714.3 million) from section 8.81. Now, I need to formally verify the total revenue for 2015 to serve as the denominator. The search results in Step 2 indicated that section 8.60 contains the “Consolidated Statements of Operations,” which is the authoritative source for revenue figures. I will read section 8.60 to extract the exact revenue amount for the year ended December 31, 2015.
</think>
<tool_call>
```
{ "arguments": "{ "section_ids": ["8.60"], "goal": "Extract the revenue amount for 2015" }", "name": "read" }
```
</tool_call>
<tool_response>
The useful information from the document section (section_id=8.60) for user goal ‘Extract the revenue amount for 2015‘ is as follows:
Evidence in document:
Heading: NETFLIX, INC.
Paragraph: CONSOLIDATED STATEMENTS OF OPERATIONS
```
<HTML_Table>
<table> ... <tr> <td>Revenues</td> <td>$ 6,779,511</td> <td>$ 5,504,656</td> ... </tr> ... </table>
</HTML_Table>
```
Caption: (in thousands, except per share data)
Summary:

Continued on next page...

Continued from previous page

Details of Case Study (Continued)
The revenue amount for 2015 is $6,779,511, as explicitly stated in the 'Revenues' row of the 'CONSOLIDATED STATEMENTS OF OPERATIONS' table for the year ended December 31, 2015. This value is presented in thousands, as indicated by the caption. </tool_response>
Final Response
Netflix's advertising expense for fiscal 2015 was **$714.3 million** [8.81], and its total revenue for the same year was **$6,779,511 thousand** (i.e., **$6,779.511 million**) [8.60]. Advertising-to-sales ratio $= \frac{714.3}{6{,}779.511} \approx 0.10536$. Rounded to three decimal places, the ratio is **0.105** (or 10.5%).

Table 3: Case study of a financial task.