DOCPILOT: INTERACTIVE AGENT ORCHESTRATION AND TRAINING FOR LONG VISUAL DOCUMENT UNDERSTANDING

Anonymous authors
Paper under double-blind review

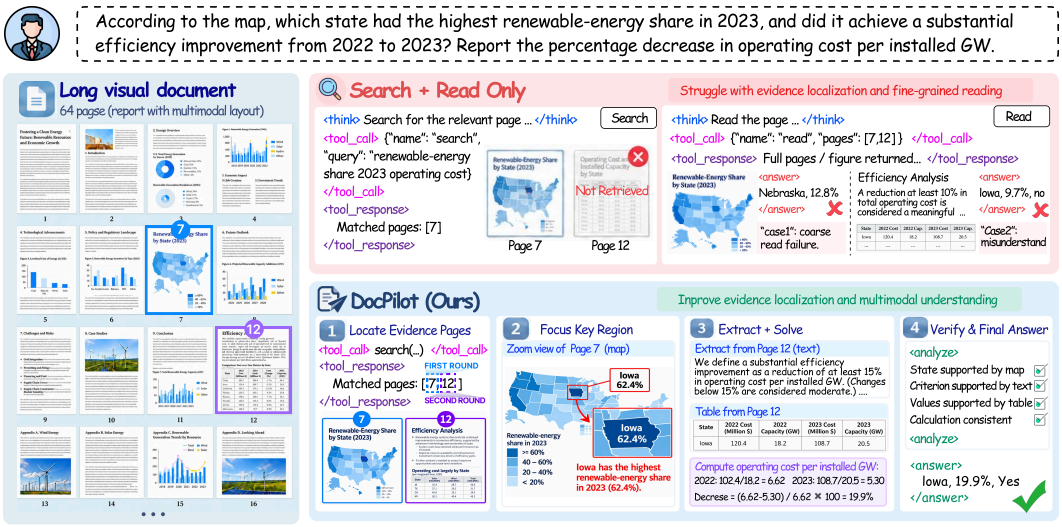

Figure 1: Illustration of DOCPILOT's capabilities in evidence seeking and multimodal understanding.

ABSTRACT

Multimodal long-document question answering requires models to seek sparse evidence across pages and modalities, understand heterogeneous document content, and reason over the collected evidence. Even advanced vision-language models (VLMs) can overlook sparse evidence and struggle with cross-modal reasoning in a single pass. Existing document agents further introduce multi-step interaction, but some still rely on predefined workflows or coarse-grained actions, limiting the flexibility and granularity of document reasoning. We introduce **DOCPILOT**, an *agentic document framework* that not only supports expressive orchestration but also makes the complex document-solving process learnable by small language models (SLMs). DOCPILOT first orchestrates the following capabilities: evidence seeking through page-level navigation and fine-grained evidence localization; and multimodal understanding through local visual inspection, code-based numerical reasoning, and cross-modal reasoning. To transfer these capabilities to small models, we propose *curriculum-based agentic training* that progresses from single-ability learning to adaptive orchestration through supervision from data-grounded reasoning traces and executable interaction trajectories. Across MMLongBench-Doc, DocBench, FinRAGBench-V, and MADQA, DOCPILOT achieves strong gains with frontier VLM backbones, while DOCPILOT-4B delivers competitive performance against substantially larger models, demonstrating strong effectiveness and generalization across diverse long-document reasoning settings.

1 INTRODUCTION

Multimodal long-document question answering (DocVQA) is a fundamental problem in real-world document intelligence, with applications in financial analysis, scientific literature, legal or compliance review, and enterprise knowledge management (Liu et al., 2025; Mathew et al., 2021). Unlike

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

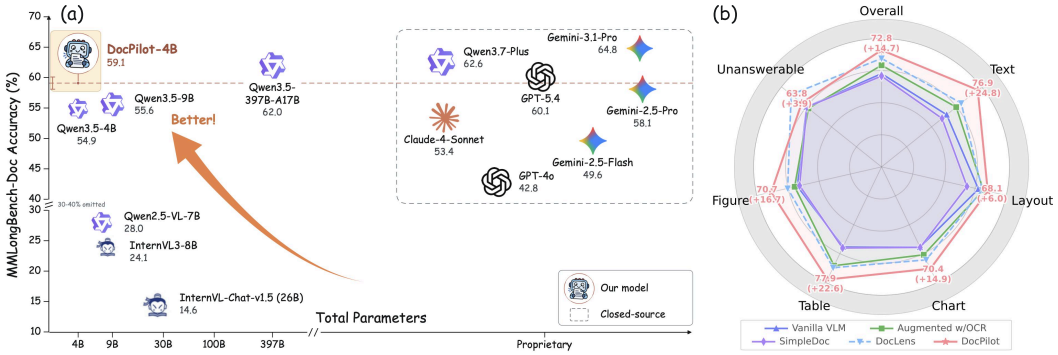

Figure 2: Performance of our proposed DOCPILOT on MMLongBench-Doc: (a) competitive accuracy with only 4B parameters; (b) performance gains with Gemini-2.5-Pro as the backbone.

text-only QA or single-image understanding, real documents contain heterogeneous information sources, including paragraphs, tables, charts, figures, and complex layouts. Answering questions over such documents often requires connecting evidence distributed across multiple pages/documents and modalities, remaining challenging even for advanced vision-language models (VLMs) (Qwen Team, 2026; ClaudeTeam, 2025; OpenAITeam, 2025; Comanici et al., 2025).

Human readers typically follow a **coarse-to-fine reading process**: they skim for a global view, selectively inspect relevant pages or regions, and revisit evidence to verify their conclusions (Pirolli & Card, 1999; Keshav, 2007; Rayner, 1998). This motivates modeling long-document QA as an iterative process supported by two core capabilities: **evidence seeking** and **multimodal understanding**. Evidence seeking identifies information needs, locates relevant evidence across documents, pages, and regions, and guides further search when evidence is insufficient. Multimodal understanding involves interpreting heterogeneous document elements and reasoning over their relationships.

Existing approaches remain limited along these two dimensions. For multimodal understanding, **OCR-based pipelines** convert documents into textual or structural representations (Xu et al., 2020), potentially losing visual cues and propagating parsing errors. **Long-context VLMs** directly process page-image sequences (Liu et al., 2026; Ma et al., 2024), but face high computational costs, context limits, and the *lost-in-the-middle* effect. **Retrieval-augmented generation (RAG)** targets evidence seeking by retrieving relevant pages or passages (Saad-Falcon et al., 2024; Cho et al., 2024; Faysse et al., 2025), but conventional static retrieval may overlook evidence dispersed across pages and modalities and lacks the flexibility to adapt to evolving information needs during reasoning.

Recent **agentic document frameworks** extend static retrieval with multi-step document interactions, but remain limited in two ways. First, predefined workflows (Han et al., 2025; Jain et al., 2025; Zhu et al., 2026) limit the flexibility to adapt reasoning paths to intermediate observations. Second, some agentic systems mainly rely on coarse-grained actions: DocDancer (Zhang et al., 2026) models document interaction mainly through SEARCH and READ, while Doc-V* (Zheng et al., 2026) adopts an OCR-free VLM-based paradigm centered on page-level retrieval and reading. As illustrated in Fig. 1, complex DocVQA requires *adaptive evidence seeking* to locate the map and supporting evidence across pages, and *fine-grained multimodal understanding* to identify the leading state, compute the reduction in cost per GW, and check the efficiency criterion. These coupled requirements motivate our agent orchestration framework, **DOCPILOT**, whose flexible composition of these operations yields substantial gains on MMLongBench-Doc (Ma et al., 2024), as shown in Fig. 2.

Despite their strong reasoning capabilities, LLM-based document agents can be costly and privacy-sensitive in practical deployment, motivating small language models (SLMs) for efficient, on-premise inference (Liang et al., 2026; Li et al., 2026). However, training compact models to master the complex orchestration required for document reasoning remains challenging. Existing approaches based on coarse-grained interactions (Zhang et al., 2026), simplify the interaction structure but leave substantial multimodal reasoning implicit within each action. Consequently, SLMs may imitate action sequences without mastering the capabilities required for reliable execution. Finer-grained actions facilitate explicit supervision of these capabilities, but increase the complexity of adaptive orchestration. Reconciling these demands in SLM-based document agents motivates our objective:

A agentic document framework should not only support expressive orchestration, but also turn the complex document-solving process into a learnable objective for SLMs.

To this end, we introduce **DOCPILOT**, a trainable interactive document agent[1] for multimodal long-document QA. We model heterogeneous document content as a unified multimodal environment, over which DOCPILOT orchestrates three complementary capabilities: (i) adaptive evidence seeking over page-level visual cues and OCR-derived structured representations, guided by intermediate state; (ii) fine-grained multimodal understanding, supported by specialized operations for local visual inspection, and code-based numerical reasoning; and (iii) iterative refinement via answer-grounding verification, enabling self-correction. As illustrated in Fig. 2 (a), with only 4B parameters, DOCPILOT achieves performance competitive with much larger proprietary models. This performance is enabled by a curriculum-based agentic training paradigm that mitigates *reward sparsity* through progressive capability learning. The curriculum proceeds in two stages: single-ability supervised fine-tuning (SFT) establishes core document interaction capabilities, while multi-ability agentic training learns to adaptively compose them according to intermediate observations. To address *trajectory scarcity*, we develop a data-grounded synthesis pipeline that provides high-quality, executable interaction trajectories for this training process. In summary, our key contributions are three-fold.

1. **Agentic Orchestration.** We introduce**DOCPILOT**, a comprehensive agent orchestration framework for long visual document QA, modeling heterogeneous documents as a unified environment for adaptive evidence seeking, fine-grained multimodal understanding, and iterative refinement.
2. **Agentic Training.** We develop a curriculum-based agentic training paradigm with data-grounded trajectory synthesis, enabling compact models to progressively acquire and orchestrate document-solving capabilities through single-ability fine-tuning and multi-ability orchestration learning.
3. **Empirical Validation.** Experiments on MMLongBench-Doc, DocBench, FinRAGBench-V, and MADQA demonstrate substantial gains across diverse baselines, while **DOCPILOT-4B** achieves competitive performance against substantially larger proprietary models with only **4B** parameters. Analyses such as evidence localization and training ablations, further substantiate robustness.

2 DOCPILOT

Fig. 3 outlines DOCPILOT, a trainable interactive document agent for multimodal long-document QA with two core components. The **agent orchestration framework** dynamically selects and composes actions for adaptive evidence seeking, multimodal understanding, and iterative refinement based on intermediate observations. The **curriculum-based agentic training** paradigm transfers these capabilities to compact models using high-quality synthesized reasoning and interaction traces, progressing from single-ability fine-tuning to multi-ability agentic training. Both components share a **multimodal document environment** for interacting with heterogeneous evidence.

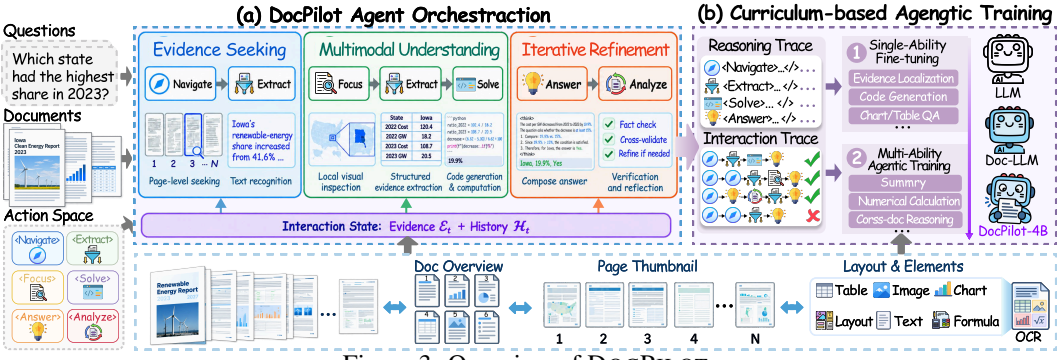

Figure 3: Overview of DOCPILOT.

2.1 MULTIMODAL DOCUMENT ENVIRONMENT

Given a document $\mathcal{D} = \{I_1, \ldots, I_N\}$, DOCPILOT constructs a multimodal environment $\mathcal{E}(\mathcal{D}) = (\{(I_i, \mathcal{S}_i, T_i)\}_{i=1}^{N}, \widetilde{\mathcal{D}})$, where I_i is the original page image, $\mathcal{S}_i$ is the OCR-derived structured representation produced by MinerU2.5 (Niu et al., 2026), T_i is a lightweight page thumbnail, and $\widetilde{\mathcal{D}}$ is the document-level overview. $\mathcal{S}_i$ preserves recognized content, layout elements, reading order, and spatial locations, while T_i is a low-resolution page thumbnail (*e.g.,* 512×512) that retains coarse layout and salient visual cues with its page number explicitly marked. The document overview

[1]An agent that can be deployed as a trainable model to interact with document environments.

further arranges consecutive thumbnails into grids to provide a compact global view, with separate overviews constructed for multi-document inputs. All representations share consistent document and page identifiers, enabling aligned access to textual, structural, and visual content.

2.2 Agent Orchestration Framework

Problem Formulation. We formulate multimodal long-document QA as an evidence-driven interaction process between the agent and a document environment. Rather than following a predefined sequence of operations or repeated sampling, DocPilot continuously updates its evidence state and selects the next action according to the unresolved information need. This yields a closed-loop process of *seeking, reasoning, verification, and refinement*, as summarized in Algorithm 1.

Given a question q and a multimodal document environment $\mathcal{D}$, a document agent typically seeks relevant evidence, interprets multimodal content, and iteratively refines its reasoning until a grounded answer is obtained. To support this process, DocPilot introduces six specialized actions, including:

- `<Navigate>`: Locate relevant visual pages.
- `<Extract>`: Extract task-relevant evidence from selected pages or regions.
- `<Focus>`: Inspect local visual regions through cropping, zooming, or subfigure focusing.
- `<Solve>`: Perform code-based numerical reasoning over collected evidence.
- `<Analyze>`: Assess answer grounding for planning, verification, and refinement.
- `<Answer>`: Produce a candidate answer.

Failed verification triggers further evidence seeking or reasoning, closing the loop through three complementary mechanisms: adaptive evidence seeking, multimodal understanding, and iterative refinement.

Algorithm 1: Inference of DocPilot

Input: Question q, environment $\mathcal{D}$, model $\mathcal{M}$
Output: Grounded answer y

Initialize evidence $\mathcal{E} \leftarrow \emptyset$ and history $\mathcal{H} \leftarrow \emptyset$;
while *answer not verified* **do**
  $(a, \phi) \leftarrow \mathcal{M}(q, \mathcal{E}, \mathcal{H})$ *// select next action*;
  if $a \in \{$ *Navigate*, *Extract*, *Focus*, *Solve* $\}$ **then**
    $o \leftarrow \mathcal{D}.\textsc{Execute}(a, \phi)$;
    $\mathcal{E} \leftarrow \textsc{Update}(\mathcal{E}, o)$;
    $\mathcal{H} \leftarrow \mathcal{H} \cup \{(a, o)\}$;
  else if $a =$ `<Answer>` **then**
    $\hat{y} \leftarrow \phi$;
    $(v, f) \leftarrow \mathcal{M}.$`<Analyze>`$(q, \mathcal{E}, \hat{y})$;
    if $v =$ `supported` **then**
      return $\hat{y}$;
    else
      $\mathcal{E} \leftarrow \textsc{Refine}(\mathcal{E}, f)$;
    end
  end
end

Adaptive Evidence Seeking. DocPilot combines `<Navigate>` and `<Extract>` to seek evidence at complementary granularities: `<Navigate>` scans page thumbnails to identify potentially relevant pages while preserving broad coverage with low visual-token cost, and `<Extract>` further operates on OCR-derived text, tables, and parsed visual elements to obtain structured, source-grounded evidence:

$$P_t = \textsc{Navigate}(q_t, \mathcal{E}_t, \{T_i\}), \qquad e_t = \textsc{Extract}(q_t, \mathcal{E}_t, P_t). \tag{1}$$

where $\{T_i\}$ denotes the page thumbnails, P_t the selected pages, and e_t the extracted structured evidence. If the evidence remains insufficient, subsequent search targets are updated from the evolving evidence state, enabling state-driven, coarse-to-fine evidence recovery.

Multimodal Understanding. To mitigate the errors introduced by directly reasoning over heterogeneous content, `<Focus>` and `<Solve>` serve distinct yet coordinated roles:

$$z_t = \textsc{Focus}(e_t; \omega_t), \qquad c_t = \textsc{Solve}(e_t),\ r_t = \textsc{ExecuteCode}(c_t). \tag{2}$$

Focus selectively applies a visual operation $\omega_t \in \{\mathrm{Crop}, \mathrm{Zoom}, \mathrm{Isolate}, \ldots\}$ to the relevant evidence e_t when finer visual inspection is required, yielding the refined observation z_t. Meanwhile, Solve generates executable code c_t for aggregation, comparison, or multi-step numerical reasoning, whose execution produces result r_t and updates the evidence state. Together, these actions enable precise visual inspection and reliable numerical reasoning over multimodal evidence.

Iterative Refinement. Once sufficient evidence has been accumulated, `<Answer>` synthesizes a candidate response, which is subsequently examined by `<Analyze>` for adequacy and faithfulness:

$$\hat{y}_t = \textsc{Answer}(q, \mathcal{E}_t), \qquad (v_t, f_t) = \textsc{Analyze}(q, \mathcal{E}_t, \hat{y}_t), \tag{3}$$

where v_t denotes the verification outcome and f_t the feedback on remaining evidence gaps or inconsistencies. A fully supported candidate is accepted; otherwise, f_t is fed back into the interaction loop to resume evidence seeking and multimodal reasoning, turning verification into an explicit mechanism for iterative self-correction. Further construction details of DocPilot, including the document environment, interaction protocol, and individual module designs, are provided in Appendix B.

2.3 CURRICULUM-BASED AGENTIC TRAINING

Data-grounded Trajectory Synthesis. Existing DocVQA data largely provide only question–answer supervision, leaving both action-level reasoning and cross-action orchestration under-specified. To support our curriculum training, we therefore synthesize two complementary forms of supervision: *reasoning traces* for learning how to execute individual capabilities, and *interaction traces* for learning when and how to compose them according to intermediate observations. Together, these traces form **DOCAGENT-INSTRUCT-40K**, including 35K single-ability reasoning traces and 5K multi-ability interaction traces for agentic training. More details are provided in Appendix C.

Reasoning Trace Synthesis. For single-ability supervision, we perform teacher-guided reasoning distillation for page-level navigation, element-level extraction, and reasoning (*e.g.,* code generation, table QA, and chart QA). We draw evidence-seeking instances from DocSeeker (Yan et al., 2026b) and HotpotQA (Yang et al., 2018), and reasoning instances from FinQA (Chen et al., 2021), ChartQAPro (Masry et al., 2025), and InfographicVQA (Mathew et al., 2022). A strong teacher distills capability-specific traces from evidence and target answers, filtered by ground-truth consistency.

Interaction Trace Synthesis. For multi-ability supervision, we draw QA instances from LongDocURL (Deng et al., 2025), DUDE (Van Landeghem et al., 2023), MADQA (Borchmann et al., 2026), and MMDocRAG (Dong et al., 2026), and roll out a strong teacher agent in the DOCPILOT environment. Conditioned on the question, interaction history, and evolving evidence state, the teacher iteratively executes actions to produce a grounded answer. These trajectories capture evidence seeking, multimodal reasoning, and verification. We retain only trajectories with correct answers, valid interactions, and well-grounded evidence, filtering out failed or redundant executions.

Single-Ability Fine-tuning. Since foundation LLMs are not explicitly trained for interactive document solving, we first strengthen the specialized capabilities required by DOCPILOT. As illustrated in Fig. 9, this stage covers page-level navigation (`<Navigate>`), element-level evidence extraction (`<Extract>`), code-based numerical reasoning (`<Solve>`), and multimodal question answering (`<Answer>`). We fine-tune the model on teacher-generated reasoning traces using next-token prediction to support subsequent orchestration learning.

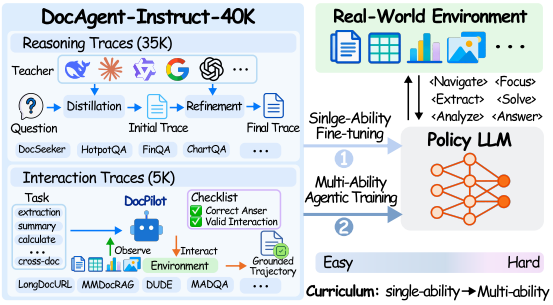

Figure 4: Curriculum-based training.

Multi-Ability Agentic Training. Building on the acquired single abilities, we further train the model on synthesized multi-ability interaction trajectories to learn adaptive orchestration for complex document solving. Each trajectory interleaves agent reasoning and actions with environment observations, culminating in a grounded answer. This supervision extends beyond isolated capability execution to tool selection, action composition, and strategy adjustment in response to intermediate evidence.

We optimize a masked autoregressive objective, retaining environment observations as conditioning context while supervising only agent-generated outputs. Given the task context c and tokenized trajectory $H = (x_1, \ldots, x_L)$, the training objective is:

$$\mathcal{L}_{\text{multi}}(\theta) = -\mathbb{E}_{(c,H)\sim\mathcal{D}_{\text{multi}}} \left[\frac{1}{\sum_{t=1}^{L} m_t} \sum_{t=1}^{L} m_t \log \pi_\theta(x_t \mid c, x_{<t}) \right]. \tag{4}$$

where $\mathcal{D}_{\text{multi}}$ denotes the interaction-trajectory training set, and m_t equals one for agent-generated reasoning, actions, and answers, and zero otherwise. Environment feedback thus informs subsequent decisions without serving as a prediction target.

3 EXPERIMENTS

We conduct experiments to answer the following research questions:

RQ1: How effective are DOCPILOT's orchestration framework and training strategy on multimodal long-document QA? **RQ2:** How much does each core component of DOCPILOT contribute to the

final performance? **RQ3:** Does DOCPILOT provide stronger evidence seeking than existing retrieval or agentic document systems? **RQ4:** How does DOCPILOT behave during interaction, and can it generalize to more challenging multi-document scenarios?

3.1 EXPERIMENTAL SETUP

Benchmarks and Datasets. We evaluate DOCPILOT on two challenging general-purpose multimodal long-document QA benchmarks: MMLongBench-Doc (Ma et al., 2024) and DocBench (Zou et al., 2025). MMLongBench-Doc contains 1,082 questions over 135 long PDFs across seven domains, averaging 47.5 pages with rich multimodal layouts, while DocBench includes 1,102 questions over 229 real-world documents spanning five domains and four major question types. For domain-specific evaluation, we use FinRAGBench-V (Zhao et al., 2025), which targets realistic financial document understanding and provides fine-grained evidence annotations for evaluating our seeking strategy. Cross-document generalization is evaluated on MADQA (Borchmann et al., 2026), a benchmark grounded in heterogeneous document collections and designed for agentic reasoning.

Metrics. We follow each benchmark's official evaluation protocol. For MMLongBench-Doc, we report rule-based accuracy and F1, while FinRAGBench-V and DocBench use LLM-as-a-judge (*LasJ*) evaluation with GPT-4o. For MADQA, we report overall, cross-page (*X-Page*), and cross-document (*X-Doc*) accuracy, with Page F1 for evidence attribution and Kuiper statistic for effort calibration.

Models and Baselines. We evaluate DOCPILOT on three baselines categories: (1) vanilla VLMs (Qwen3.5-4B/9B/397B-A17B, GPT-4o, GPT-5.4 (OpenAITeam, 2025), Qwen3.7-Plus (Qwen Team, 2026), Claude-4-Sonnet (ClaudeTeam, 2025), and Gemini-2.5/3.1 models); (2) OCR-based methods using Claude-4-Sonnet and Gemini-2.5-Flash/Pro (Comanici et al., 2025); and (3) Agentic frameworks including M3DocRAG (Cho et al., 2024), MDocAgent (Han et al., 2025), DocDancer (Zhang et al., 2026), SimpleDoc (Jain et al., 2025), and DocLens (Zhu et al., 2026). On MADQA, we additionally compare with OpenAI File Search, HEAVEN (Kim et al., 2026), Codex, BM25 Agent, and Recursive Language Models (RLM) (Zhang et al., 2025a), all with GPT-5.4 as the backbone.

Implementation Details. DOCPILOT adopt the vanilla ReAct (Yao et al., 2022) as the agent's framework. We implement curriculum-based training in ms-swift[2], comprising single-ability SFT followed by multi-ability agentic training. Both stages use 3 training epochs, a learning rate of 10^{-4}, a batch size of 64, and LoRA adapters with rank $r = 16$ and $\alpha = 32$. The maximum sequence lengths are 8K and 24K text tokens for the two stages, respectively, excluding visual tokens.

3.2 MAIN RESULTS

Table 1 summarizes the main results across MMLongBench-Doc, FinRAGBench-V, and DocBench.

Large gains across LLMs. Despite the strong standalone performance of modern VLM backbones, DOCPILOT delivers substantial gains across all evaluated models. On the general-purpose benchmarks, DOCPILOT improves vanilla backbones by 10.1, 14.7, 10.8, and 13.4 points on MMLongBench-Doc, and 3.7, 9.3, 6.2, and 10.0 points on DocBench for Qwen3.5-397B-A17B, Gemini-2.5-Pro, Gemini-3.1-Pro, and GPT-5.4, respectively. On FinRAGBench-V, the corresponding gains reach 9.8, 18.5, 11.5, and 18.7 LasJ points. These results demonstrate that DOCPILOT complements strong base models, with consistent benefits across backbones and benchmarks.

Advantages over existing agentic frameworks. DOCPILOT also consistently outperforms prior document agents under comparable backbones. With Gemini-2.5-Pro, DOCPILOT improves MMLongBench-Doc accuracy from 67.6 with DocLens to 72.8, and FinRAGBench-V LasJ from 70.4 to 73.4. On DocBench, it reaches 84.1 LasJ with the same backbone, exceeding DocDancer's 79.9, highlighting the effectiveness of DOCPILOT's richer orchestration over existing agentic pipelines.

Robustness across multimodal capabilities. A finer-grained breakdown reveals three complementary capability patterns. First, semantic and structural understanding improves substantially: TXT gains +24.8 with Gemini-2.5-Pro and +10.4 with GPT-5.4, while LAY increases by +6.0 and +6.2, respectively. Second, gains are strongest on visually grounded reasoning. Gemini-2.5-Pro raises CHA/TAB/FIG by +14.9/+22.6/+16.7 on MMLongBench-Doc, while GPT-5.4 delivers +24.5/+35.7 on CHA/TAB on FinRAGBench-V. Finally, answer reliability is markedly strengthened,

[2] https://github.com/modelscope/ms-swift

Table 1: Performance comparison on multimodal long-document QA benchmarks. Results are reported by evidence source: text (TXT), layout (LAY), chart (CHA), table (TAB), figure (FIG), and unanswerable (UNA), with overall accuracy and F1. † denotes the reused result from prior papers. Blue highlights the best, while Red indicates the second-best.

Model	MMLongBench-Doc								FinRAGBench-V				DocBench
	TXT	LAY	CHA	TAB	FIG	UNA	Acc	F1	TXT	TAB	CHA	LasJ	LasJ
Vanilla VLMs													
Open-source Models													
Qwen3.5-4B	54.6	57.7	50.8	57.1	47.2	54.7	54.9	53.5	42.4	50.4	65.8	52.7	69.3
Qwen3.5-9B	55.6	56.3	47.5	52.5	47.9	60.1	55.6	55.2	43.1	55.5	58.1	52.9	69.8
Qwen3.5-397B-A17B	65.1	73.9	66.1	72.3	60.8	36.8	62.0	63.3	52.1	64.7	69.0	62.3	80.6
Proprietary Models													
GPT-4o	46.3	46.0	45.3	50.0	44.1	20.2	42.8	44.9	53.9	33.3	52.6	43.4	63.1
GPT-5.4	62.3	62.2	65.0	75.2	58.6	31.8	60.1	61.8	59.0	49.3	60.0	55.1	73.5
Qwen3.7-Plus	67.7	76.5	64.4	75.5	69.0	26.9	62.6	63.7	55.6	69.8	69.7	65.9	83.2
Claude-4-Sonnet	50.4	49.4	50.5	57.3	43.9	59.0	53.4	–	36.6	20.2	51.9	33.8	–
Gemini-2.5-Flash	44.0	53.2	46.0	43.9	48.2	56.7	49.6	–	49.0	41.6	41.0	43.0	–
Gemini-2.5-Pro	52.1	62.1	55.5	55.3	54.0	59.9	58.1	–	62.2	55.3	50.4	54.9	74.8
Gemini-3.1-Pro	68.0	73.9	66.1	74.3	65.5	39.9	64.8	66.6	50.0	72.7	64.5	64.3	80.6
VLMs Augmented with OCR													
Claude-4-Sonnet	52.7	51.6	50.0	58.1	45.3	65.9	56.0	–	58.7	21.6	54.3	41.0	–
Gemini-2.5-Flash	55.9	54.9	52.7	63.4	50.3	60.8	58.5	–	67.6	64.4	46.1	58.3	–
Gemini-2.5-Pro	59.7	65.3	60.8	68.3	55.7	58.4	63.3	–	70.0	70.0	56.2	64.9	–
VLM-based Agentic Frameworks													
M3DocRAG (w/ Qwen2-VL-7B)†	30.0	23.5	18.9	20.1	20.8	5.8	21.0	–	–	–	–	–	–
MDocAgent (w/ GPT-4o)†	–	–	–	–	–	–	42.0	–	–	–	–	–	–
DocDancer (FT w/ Qwen3-4B)†	–	–	–	–	–	–	48.4	49.2	–	–	–	–	79.8
DocDancer (w/ Gemini-2.5-Pro)†	–	–	–	–	–	–	56.3	55.3	–	–	–	–	79.9
SimpleDoc													
w/ Claude-4-Sonnet	52.1	53.3	58.3	62.4	46.9	66.5	58.6	–	59.6	68.9	54.9	61.7	–
w/ Gemini-2.5-Flash	45.5	57.4	49.0	51.6	45.2	66.5	53.3	–	70.2	56.2	53.6	58.3	–
w/ Gemini-2.5-Pro	48.4	54.8	55.7	56.1	52.5	59.7	56.6	–	67.5	64.0	60.9	63.6	–
DocLens†													
w/ Claude-4-Sonnet	59.9	58.2	54.4	63.9	55.3	74.0	63.3	–	70.2	66.0	60.3	64.8	–
w/ Gemini-2.5-Flash	59.5	61.5	54.8	66.9	59.0	73.8	64.7	–	69.9	71.3	64.5	68.5	–
w/ Gemini-2.5-Pro	63.7	64.6	64.3	69.7	60.2	72.2	67.6*	–	68.9	74.2	67.1	70.4	–
DocPilot (Ours)													
w/ Qwen3.5-397B-A17B	71.8	52.1	65.5	72.5	63.3	83.0	72.1	68.8	78.6	66.8	74.0	72.1	84.3
w/ Gemini-2.5-Pro	76.9	68.1	70.4	77.9	70.7	63.8	72.8	73.0	72.0	72.6	75.8	73.4	84.1
w/ Gemini-3.1-Pro	75.4	70.0	74.6	74.5	74.2	73.2	75.6	74.9	68.3	79.4	77.4	75.8	86.8
w/ GPT-5.4	72.7	68.4	69.1	71.6	71.4	75.0	73.5	72.4	59.3	79.4	78.7	73.8	83.5
DocPilot-4B	57.4(+2.8)	47.9(-9.8)	53.4(+2.6)	58.7(+1.6)	51.0(+3.8)	70.0(+15.3)	59.1(+4.2)	58.5(+5.0)	68.7(+26.3)	56.2(+5.8)	59.6(-6.2)	60.2(+7.5)	80.3(+11.0)
Human Expert†	–	–	–	–	–	–	65.8	66.0	–	–	–	–	–

with Unanswerable (UNA) rising from 31.8 to 75.0 for GPT-5.4 and from 39.9 to 73.2 for Gemini-3.1-Pro. Overall, DocPilot exhibits strong and balanced capabilities across semantic understanding, fine-grained multimodal reasoning, and evidence-grounded answerability.

Compact agentic models ≈ LLM. DocPilot-4B consistently outperforms vanilla Qwen3.5-4B, with absolute gains of 4.2%, 7.5%, and 11.0% on MMLongBench-Doc, FinRAGBench-V, and DocBench, respectively. These improvements support the effectiveness of curriculum-based training in developing and coordinating document-solving capabilities. With only 4B parameters, it surpasses vanilla Gemini-2.5-Pro across all three benchmarks and GPT-5.4 on the latter two, while nearly matching Gemini-3.1-Pro on DocBench (80.3 vs. 80.6). These results highlight learned document interaction as a route to competitive performance at a compact model scale.

3.3 Ablation Studies

We conduct ablation studies to understand which components are most important for DocPilot. The analysis is divided into two parts: ablation on core actions and ablation on the training strategy.

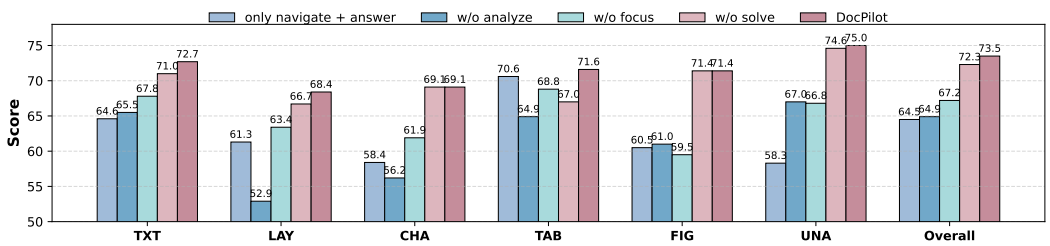

Figure 5: Ablation of core tools in DocPilot on MMLongBench-Doc.

Ablation on Core Modules. Fig. 5 presents ablations on MMLongBench-Doc using GPT-5.4 as the backbone, highlighting the complementary roles of specialized tools across evidence sources. Removing <Focus> reduces chart and figure accuracy by 7.2 and 11.9 points, respectively, demonstrating the importance of local visual inspection. The 4.6% drop in table accuracy without <Solve> supports code-based numerical reasoning. Ablating <Analyze> reduces accuracy from 73.5 to 64.9, leaving only a 0.4% advantage over NAVIGATE+ANSWER despite the additional tools. This suggests that fundamental tools can yield meaningful gains, whereas realizing the benefits of a richer tool set requires adaptive orchestration that aligns tool selection and composition with intermediate observations.

Ablation on Training Strategy. Table 2 shows that curriculum-based training outperforms direct multi-ability SFT by 2.3 points on MMLongBench-Doc and 1.8 points on FinRAGBench-V. The 4.0-point gain on answerable questions suggests that developing individual capabilities first improves their subsequent orchestration. The accompanying 4.4-point decline on unanswerable questions may reflect a greater tendency to answer beyond the available evidence, motivating stronger grounding verification. These gains coincide with a shift in tool use from broad navigation toward fine-grained evidence inspection; see Appendix D.2 for details.

Table 2: Ablation study of different training strategies.

Methods	MMLong			FinRAG
	ANS	UNA	ALL	
DOCPILOT-4B	**56.3**	70.0	**59.1**	**60.2**
– Only Multi-ability SFT	52.3	**74.4**	56.8	58.4

3.4 BEHAVIOR AND ROBUSTNESS ANALYSIS

Evidence Localization Performanc. Table 3 reports page-level evidence localization performance. Compared with conventional retrievers, BM25 and ColQwen2.5 (Faysse et al., 2025), DocLens raises page recall to 97.3 through repeated sampling with Gemini-2.5-Pro, retrieving 3.5 pages on average at 55.1 precision. Under the same backbone, DOCPILOT retrieves only 3.0 pages on average and raises precision to 73.2 while maintaining 91.8 recall, improving final QA accuracy from 67.6 to 72.8. With GPT-5.4, DOCPILOT also reaches 95.0 recall and 73.5 accuracy.

Fig. 6 evaluates element-level localization on FinRAGBench-V using annotated evidence boxes. DOCPILOT improves precision, recall, and F1 over DocLens by +6.8, +0.8, and +4.5 points, respectively, showing that local inspection better matches the reasoning need than exhaustively exposing detected elements. Together with state-conditioned page navigation, this makes evidence localization adaptive and interleaved with reasoning, with each step guiding further evidence acquisition.

Table 3: Page localization on MMLongBench-Doc.

Method	#Pages↓	Rec.↑	Prec.↑	Acc.↑
BM25 Top-5	5	48.5	14.7	–
ColQwen2.5 Top-5	5	74.8	23.5	–
ColQwen2.5 Top-10	10	83.9	14.5	–
MDocAgent Retriever	13.6	71.1	7.0	49.6
SimpleDoc Retriever	4.9	89.0	34.7	64.0
DocLens (Gemini-2.5-Pro)	3.5	**97.3**	55.1	67.6
DOCPILOT (GPT-5.4)	4.3	95.0	55.8	**73.5**
DOCPILOT (Gemini-2.5-Pro)	**3.0**	91.8	**73.2**	72.8

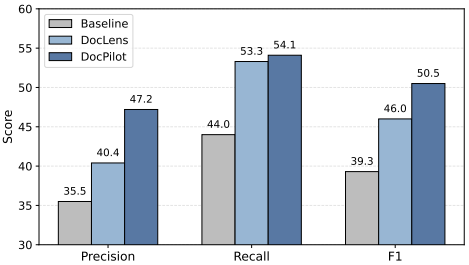

Figure 6: Element localization evaluation

Tool-Use Analysis. Fig. 7 reveals distinct orchestration patterns across backbones. Navigate remains consistently frequent (2.6–3.0 calls), suggesting that most behavioral variation emerges after localization. Gemini-3.1-Pro yields the shortest trajectories (7.9 actions), whereas Qwen3.7-397B-A 17B produces the longest (11.4), with heavier <Extract> and <Solve> usage. Actions are also selectively invoked: GPT-5.4 rarely invokes <Solve> (0.1), instead favoring <Focus> for fine-grained visual inspection. Meanwhile, repeated action calls highlight iterative evidence acquisition, inspection, and verification, while overall usage patterns reveal backbone-specific orchestration strategies.

Cross-Document Generalization on MADQA. We further evaluate DOCPILOT on the 200-sample MADQA development set under an oracle-guided setting, where relevant documents are provided directly, isolating evidence localization from document retrieval. With GPT-5.4, DOCPILOT reaches 86.5 accuracy, comparable to Codex (86.7) while surpassing the BM25 Agent (61.5) and RLM (60.5). It performs particularly well on cross-document questions, achieving the best X-Doc score of 89.5, alongside 59.5 X-Page and 75.0 Page F1. Moreover, DOCPILOT records the lowest Kuiper score

Table 4: Performance of different systems on MADQA.

Model / Framework	Accuracy	X-Page	X-Doc	Page F1	Kuiper ↓
Non-Agentic Systems					
GPT-5.4 File Search	65.1 ± 6.8	36.8 ± 22.6	26.9 ± 20.2	64.5	–
GPT-5.4 HEAVEN	71.9 ± 6.4	53.9 ± 23.5	57.4 ± 24.8	61.9	–
GPT-5.4 Codex	**86.7 ± 5.0**	**79.4 ± 19.6**	80.6 ± 18.7	**83.0**	–
Agentic Systems					
GPT-5.4 BM25 Agent	61.5 ± 6.9	48.2 ± 23.5	48.3 ± 22.9	58.3	16.7
GPT-5.4 RLM	60.5 ± 6.9	56.7 ± 23.4	63.8 ± 24.2	22.1	22.9
GPT-5.4 DocPilot	86.5 ± 5.1	59.5 ± 23.2	**89.5 ± 11.4**	75.0	**5.7**
DocPilot-4B	60.0 ± 7.0	39.7 ± 23.0	40.3 ± 22.4	50.1	10.3

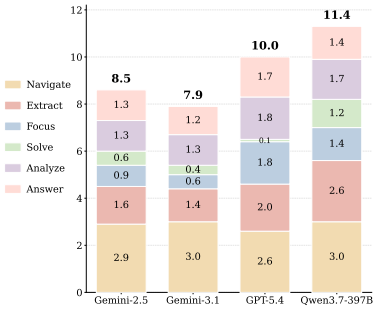

Figure 7: Average action calls.

(5.7) among agentic systems, indicating better effort calibration. Overall, these results show strong generalization to multi-document reasoning, especially for integrating evidence across documents.

4 Related Work

4.1 Multimodal Long-Document Reasoning

Early approaches derive textual or structural representations using OCR or parsers, potentially losing critical visual cues (Xu et al., 2020). Vision-language models (VLMs) process page images directly but may overlook sparse evidence in long contexts and produce hallucinations (Ma et al., 2024). Retrieval-augmented methods (Dong et al., 2026; Zhang et al., 2024) scale by retrieving relevant text (Chen et al., 2024; Santhanam et al., 2022) or visual pages (Faysse et al., 2025) before reasoning (e.g., ViDoRAG (Wang et al., 2025a), M3DocRAG (Cho et al., 2024), and VRAG-RL (Wang et al., 2025b)). Yet their static, similarity-driven retrieval may miss evidence required for reasoning.

Agentic frameworks enable multi-step document interaction. MDocAgent (Han et al., 2025) routes evidence through specialized agents; SimpleDoc (Jain et al., 2025) alternates dual-cue retrieval with query refinement; and DocLens (Zhu et al., 2026) follows a coarse-to-fine pipeline of repeated page localization and exhaustive element enrichment. More flexible approaches include DocDancer (Zhang et al., 2026), which iteratively seeks information through Search and Read, and Doc-V* (Zheng et al., 2026), which aggregates page-level evidence without OCR. However, prescribed workflows or limited action spaces often constrain adaptive orchestration. DocPilot instead coordinates fine-grained actions for evidence seeking, multimodal understanding, and iterative refinement.

4.2 Trainable Document Agents

Recent studies train document agents' interaction policies from trajectories and environment feedback (Zhang et al., 2025b; Yan et al., 2026a; Wang et al., 2025b; Jin et al., 2025). With limited agentic supervision in DocQA, synthetic trajectories offer a scalable alternative to manual annotation (Liu et al., 2025; Qwen Team, 2026). DocDancer (Zhang et al., 2026) learns Search/Read behaviors from synthesized trajectories, while Doc-V* (Zheng et al., 2026) combines trajectory imitation with reinforcement learning for page navigation. However, coarse-grained trajectories often weakly supervise multimodal capabilities, whereas fine-grained actions complicate learning adaptive composition. DocPilot addresses this tension through data-grounded trajectory synthesis and curriculum-based agentic training, progressing from single-ability acquisition to multi-ability orchestration.

5 Conclusion and Future Work

In this work, we presented DocPilot, an agentic document framework that unifies adaptive evidence seeking, multimodal understanding, and iterative refinement within a shared document environment. Data-grounded trajectory synthesis supports a curriculum from single-ability fine-tuning to multi-ability agentic training, enabling compact models to learn both individual capabilities and their adaptive composition. Evaluations on four benchmarks show substantial gains across strong VLM backbones and improved evidence localization. Ablations further highlight the importance of orchestrating complementary tools to coordinate evidence seeking and reasoning.

Looking ahead, DocPilot provides a foundation for autonomous, adaptive document intelligence. We will explore self-evolution by distilling successful strategies and failure lessons from accumulated trajectories, alongside an evolving semantic layer that consolidates domain knowledge across tasks.

AI USE STATEMENT

LLMs are used only for auxiliary purposes such as language refinement, minor code debugging, synthetic data construction, and experiment evaluation support. They do not contribute to the research design, methodology, or core writing of the paper. Accordingly, LLM usage does not constitute a substantive contribution to the intellectual content of this work.

ETHICS STATEMENT

Our experiments use publicly available document QA datasets. DOCPILOT advances evidence-grounded document understanding, while its compact model supports resource-efficient, on-premise deployment. Generated answers may contain errors and require verification against source evidence, particularly in high-stakes domains such as finance and law. Deployments involving sensitive documents should incorporate appropriate access controls and privacy safeguards.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide the implementation of DOCPILOT, including document preprocessing, agent orchestration, trajectory synthesis, and evaluation scripts, at `https://anonymous.4open.science/r/DocPilot`. The appendix details data construction procedures, and evaluation protocols for reproducing our experiments.

REFERENCES

Łukasz Borchmann, Jordy Van Landeghem, Michał Turski, Shreyansh Padarha, Ryan Othniel Kearns, Adam Mahdi, Niels Rogge, Clémentine Fourrier, Siwei Han, Huaxiu Yao, et al. Strategic navigation or stochastic search? how agents and humans reason over document collections. *arXiv preprint arXiv:2603.12180*, 2026.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. BGE m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *CoRR*, abs/2402.03216, 2024. doi: 10.48550/ARXIV.2402.03216. URL `https://doi.org/10.48550/arXiv.2402.03216`.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan R Routledge, et al. Finqa: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, 2021.

Jaemin Cho, Debanjan Mahata, Ozan Irsoy, Yujie He, and Mohit Bansal. M3docrag: Multi-modal retrieval is what you need for multi-page multi-document understanding. *CoRR*, abs/2411.04952, 2024. doi: 10.48550/ARXIV.2411.04952. URL `https://doi.org/10.48550/arXiv.2411.04952`.

ClaudeTeam. Claude sonnet 4: Hybrid reasoning model with superior intelligence for high-volume use cases, and 200k context window, 2025. URL `https://www.anthropic.com/claude/sonnet`.

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

Chao Deng, Jiale Yuan, Pi Bu, Peijie Wang, Zhong-Zhi Li, Jian Xu, Xiao-Hui Li, Yuan Gao, Jun Song, Bo Zheng, et al. 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)*, pp. 1135–1159, 2025.

Kuicai Dong, CHANG YUJING, Shijie Huang, Yasheng Wang, Ruiming Tang, and Yong Liu. Benchmarking retrieval-augmented multimomal generation for document question answering. *Advances in Neural Information Processing Systems*, 38, 2026.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. Colpali: Efficient document retrieval with vision language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=ogjBpZ8uSi.

Siwei Han, Peng Xia, Ruiyi Zhang, Tong Sun, Yun Li, Hongtu Zhu, and Huaxiu Yao. Mdocagent: A multi-modal multi-agent framework for document understanding. *CoRR*, abs/2503.13964, 2025. doi: 10.48550/ARXIV.2503.13964. URL https://doi.org/10.48550/arXiv.2503.13964.

Chelsi Jain, Yiran Wu, Yifan Zeng, Jiale Liu, Shengyu Dai, Zhenwen Shao, Qingyun Wu, and Huazheng Wang. Simpledoc: Multi-modal document understanding with dual-cue page retrieval and iterative refinement. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 28410–28427. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1443. URL https://doi.org/10.18653/v1/2025.emnlp-main.1443.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *CoRR*, abs/2503.09516, 2025. doi: 10.48550/ARXIV.2503.09516. URL https://doi.org/10.48550/arXiv.2503.09516.

Srinivasan Keshav. How to read a paper. *ACM SIGCOMM Computer Communication Review*, 37(3): 83–84, 2007.

Juyeon Kim, Geon Lee, Dongwon Choi, Taeuk Kim, and Kijung Shin. Hybrid-vector retrieval for visually rich documents: Combining single-vector efficiency and multi-vector accuracy. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 1073–1089, 2026.

Boyan Li, Zhuowen Liang, Yupeng Xie, Xiaotian Lin, Tianqi Luo, Xinyu Liu, Yizhang Zhu, Zhangyang Peng, Yuan Li, Zhengxuan Zhang, et al. Dataspace: Benchmarking data agents for verifiable analytics over heterogeneous workspaces. *arXiv preprint arXiv:2608.03451*, 2026.

Zhuowen Liang, Xiaotian Lin, Zhengxuan Zhang, Yuyu Luo, Haixun Wang, and Nan Tang. Long-document qa with chain-of-structured-thought and fine-tuned slms. *arXiv preprint arXiv:2603.29232*, 2026.

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

Yuliang Liu, Biao Yang, Qiang Liu, Zhang Li, Zhiyin Ma, Shuo Zhang, and Xiang Bai. Textmonkey: An ocr-free large multimodal model for understanding document. *IEEE Trans. Pattern Anal. Mach. Intell.*, 48(5):6008–6019, 2026. doi: 10.1109/TPAMI.2026.3653415. URL https://doi.org/10.1109/TPAMI.2026.3653415.

Yubo Ma, Yuhang Zang, Liangyu Chen, Meiqi Chen, Yizhu Jiao, Xinze Li, Xinyuan Lu, Ziyu Liu, Yan Ma, Xiaoyi Dong, Pan Zhang, Liangming Pan, Yu-Gang Jiang, Jiaqi Wang, Yixin Cao, and Aixin Sun. MMLONGBENCH-DOC: benchmarking long-context document understanding with visualizations. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/ae0e43289bffea0c1fa34633fc608e92-Abstract-Datasets_and_Benchmarks_Track.html.

Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, et al. Chartqapro: A more diverse and challenging benchmark for chart question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19123–19151, 2025.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. Docvqa: A dataset for VQA on document images. In *IEEE Winter Conference on Applications of Computer Vision, WACV 2021, Waikoloa, HI, USA, January 3-8, 2021*, pp. 2199–2208. IEEE, 2021. doi: 10.1109/WACV48630.2021.00225. URL https://doi.org/10.1109/WACV48630.2021.00225.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2582–2591. IEEE, 2022.

Junbo Niu, Zheng Liu, Zhuangcheng Gu, Bin Wang, Linke Ouyang, Zhiyuan Zhao, Tao Chu, Tianyao He, Fan Wu, Qintong Zhang, et al. Mineru2. 5: A decoupled vision-language model for efficient high-resolution document parsing. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL 2026)*, pp. 13–42, 2026.

OpenAITeam. Introducing gpt-5, 2025. URL https://openai.com/index/introducing-gpt-5/.

Peter Pirolli and Stuart K. Card. Information foraging. *Psychological Review*, 106(4):643–675, 1999.

Qwen Team. Qwen3.7-Plus: Multimodal agent intelligence, 2026. URL https://qwen.ai/blog?id=qwen3.7-plus.

Keith Rayner. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, 124(3):372–422, 1998.

Jon Saad-Falcon, Joe Barrow, Alexa F. Siu, Ani Nenkova, Seunghyun Yoon, Ryan A. Rossi, and Franck Dernoncourt. Pdftriage: Question answering over long, structured documents. In Franck Dernoncourt, Daniel Preotiuc-Pietro, and Anastasia Shimorina (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: EMNLP 2024 - Industry Track, Miami, Florida, USA, November 12-16, 2024*, pp. 153–169. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.EMNLP-INDUSTRY.13. URL https://doi.org/10.18653/v1/2024.emnlp-industry.13.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pp. 3715–3734. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.NAACL-MAIN.272. URL https://doi.org/10.18653/v1/2022.naacl-main.272.

Jordy Van Landeghem, Rubèn Tito, Łukasz Borchmann, Michał Pietruszka, Pawel Joziak, Rafal Powalski, Dawid Jurkiewicz, Mickaël Coustaty, Bertrand Anckaert, Ernest Valveny, et al. Document understanding dataset and evaluation (dude). In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 19528–19540, 2023.

Qiuchen Wang, Ruixue Ding, Zehui Chen, Weiqi Wu, Shihang Wang, Pengjun Xie, and Feng Zhao. Vidorag: Visual document retrieval-augmented generation via dynamic iterative reasoning agents. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 9113–9134. Association for Computational Linguistics, 2025a. doi: 10.18653/V1/2025.EMNLP-MAIN.464. URL https://doi.org/10.18653/v1/2025.emnlp-main.464.

Qiuchen Wang, Ruixue Ding, Yu Zeng, Zehui Chen, Lin Chen, Shihang Wang, Pengjun Xie, Fei Huang, and Feng Zhao. VRAG-RL: empower vision-perception-based RAG for visually rich information understanding via iterative reasoning with reinforcement learning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025b. URL http://papers.nips.cc/paper_files/paper/2025/hash/529b876b5d87f997e98ed925a3ce9f01-Abstract-Conference.html.

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

Hao Yan, Yuliang Liu, Xingchen Liu, Yuyi Zhang, Minghui Liao, Jihao Wu, Wei Chen, and Xiang Bai. Docseeker: Structured visual reasoning with evidence grounding for long document understanding. *CoRR*, abs/2604.12812, 2026a. doi: 10.48550/ARXIV.2604.12812. URL https://doi.org/10.48550/arXiv.2604.12812.

Hao Yan, Yuliang Liu, Xingchen Liu, Yuyi Zhang, Minghui Liao, Jihao Wu, Wei Chen, and Xiang Bai. Docseeker: Structured visual reasoning with evidence grounding for long document understanding. *arXiv preprint arXiv:2604.12812*, 2026b.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Alex L Zhang, Tim Kraska, and Omar Khattab. Recursive language models. *arXiv preprint arXiv:2512.24601*, 2025a.

Jinxu Zhang, Yongqi Yu, and Yu Zhang. CREAM: coarse-to-fine retrieval and multi-modal efficient tuning for document VQA. In Jianfei Cai, Mohan S. Kankanhalli, Balakrishnan Prabhakaran, Susanne Boll, Ramanathan Subramanian, Liang Zheng, Vivek K. Singh, Pablo César, Lexing Xie, and Dong Xu (eds.), *Proceedings of the 32nd ACM International Conference on Multimedia, MM 2024, Melbourne, VIC, Australia, 28 October 2024 - 1 November 2024*, pp. 925–934. ACM, 2024. doi: 10.1145/3664647.3680750. URL https://doi.org/10.1145/3664647.3680750.

Qintong Zhang, Xinjie Lv, Jialong Wu, Baixuan Li, Zhengwei Tao, Guochen Yan, Huanyao Zhang, Bin Wang, Jiahao Xu, Haitao Mi, and Wentao Zhang. Docdancer: Towards agentic document-grounded information seeking. *CoRR*, abs/2601.05163, 2026. doi: 10.48550/ARXIV.2601.05163. URL https://doi.org/10.48550/arXiv.2601.05163.

Shaolei Zhang, Ju Fan, Meihao Fan, Guoliang Li, and Xiaoyong Du. Deepanalyze: Agentic large language models for autonomous data science. *CoRR*, abs/2510.16872, 2025b. doi: 10.48550/ARXIV.2510.16872. URL https://doi.org/10.48550/arXiv.2510.16872.

Suifeng Zhao, Zhuoran Jin, Sujian Li, and Jun Gao. Finragbench-v: A benchmark for multimodal rag with visual citation in the financial domain. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4215–4249, 2025.

Yuanlei Zheng, Pei Fu, Hang Li, Ziyang Wang, Yuyi Zhang, Wenyu Ruan, Xiaojin Zhang, Zhongyu Wei, Zhenbo Luo, Jian Luan, Wei Chen, and Xiang Bai. Doc-vˆ*: Coarse-to-fine interactive visual reasoning for multi-page document VQA. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 45901–45923. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.2129. URL https://doi.org/10.18653/v1/2026.acl-long.2129.

Dawei Zhu, Rui Meng, Jiefeng Chen, Sujian Li, Tomas Pfister, and Jinsung Yoon. Doclens: A tool-augmented multi-agent framework for long visual document understanding. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 26804–26829. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.1234. URL https://doi.org/10.18653/v1/2026.acl-long.1234.

Table 5: Statistics and characteristics of the evaluation benchmarks.

Benchmarks	Document		Question		Evi. Loc.		Doc. Input	
	Domain	#Pages	#Num	Expert	Page	Quote	Single	Multi
MMLongBench-Doc	Multiple	47.5	1,082	✓	✓	✗	✓	✗
DocBench	Multiple	66.0	1,102	✓	✗	✗	✓	✗
FinRAGBench-V	Financial	50.7	1,394	✓	✓	✗	✓	✗
MADQA	Multiple	23.3	2,250	✓	✓	✗	✓	✓

Anni Zou, Wenhao Yu, Hongming Zhang, Kaixin Ma, Deng Cai, Zhuosheng Zhang, Hai Zhao, and Dong Yu. 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*, pp. 359–373, 2025.

A Dataset Statistics

We evaluate DocPilot on four benchmarks spanning three document reasoning settings: general-purpose understanding, domain-specific financial analysis, and cross-document reasoning. Table 5 summarizes their key statistics.

General-Purpose Understanding. We adopt MMLongBench-Doc (Ma et al., 2024) and DocBench (Zou et al., 2025) as two challenging general-purpose benchmarks. MMLongBench-Doc contains 135 long PDFs across seven document types, averaging 47.5 pages and 21,214 textual tokens. Its refined evaluation set includes 1,082 questions, with 372 cross-page and 223 unanswerable cases. The benchmark spans text, tables, charts, figures, and layout structures, providing a comprehensive testbed for long-context evidence seeking and multimodal reasoning.

DocBench contains 229 real-world PDFs and 1,102 questions across five domains: Academia, Finance, Government, Law, and News. The documents average 66 pages and 46,377 tokens, while the questions span text-only, multimodal, metadata, and unanswerable settings. It further exhibits substantial document-length variation, from single-page news articles to financial reports averaging 192 pages. Together, MMLongBench-Doc and DocBench provide complementary coverage of general long-document understanding across diverse layouts, modalities, and reasoning requirements.

Domain-Specific Financial Analysis. We further evaluate on FinRAGBench-V (Zhao et al., 2025), a multimodal benchmark built from real-world financial documents. Its retrieval corpus contains 2,209 documents and 111,999 pages spanning financial reports, statements, prospectuses, academic papers, magazines, and news. We evaluate on the 539 English QA pairs, covering text inference, chart understanding, numerical calculation, table reasoning, comparison, and multi-page queries. Its dense financial tables, charts, and fine-grained visual evidence make it well suited for evaluating precise evidence localization and domain-specific multimodal reasoning.

Cross-Document Reasoning. Finally, we use the 200-question development set of MADQA (Borchmann et al., 2026) to evaluate reasoning over document collections. Unlike single-document benchmarks, MADQA requires agents to navigate document collections and aggregate dispersed evidence, making it well suited for evaluating cross-document evidence seeking and reasoning.

B Construction of DocPilot

B.1 Detail of Environment Design

This subsection provides additional construction details for the multimodal document environment introduced in Sec. 2.1, including the OCR-derived structure, page thumbnails, and document overview.

OCR-derived Structured Representation. We employ MinerU2.5 (Niu et al., 2026) for OCR and layout parsing, representing each page as structured document elements rather than flattened OCR text. For page I_i, the resulting representation $\mathcal{S}_i = \{s_{i,1}, \ldots, s_{i,M_i}\}$ contains elements such as titles, paragraphs, tables, figures, captions, and footnotes. Each element retains its recognized content,

semantic type, reading order, bounding box, and page index, preserving both the semantic content and spatial structure of the original page. Structurally irrelevant elements, such as headers and footers, are removed during parsing.

Page Thumbnail Representation. For each page I_i, the environment maintains both a high-resolution page screenshot and a lightweight thumbnail T_i. The page screenshot is capped at 1024×768 pixels, while T_i is resized to 512×512 and annotated with its absolute page number. Although detailed text may be difficult to read in the thumbnail, it preserves coarse visual cues such as headings, layout, and the distribution of tables, charts, and figures, while the original screenshot supports subsequent fine-grained extraction and inspection.

Document Overview Representation. To provide a compact document-level representation, we further construct a document overview $\widetilde{\mathcal{D}}$ from the page thumbnails. Consecutive thumbnails are arranged in reading order into a regular grid, with their page indices retained for unambiguous correspondence with the original document. For documents that exceed the capacity of a single grid, the thumbnails are partitioned into consecutive groups and represented by multiple overview images. For multi-document inputs, an overview is constructed independently for each document and annotated with its document identifier and title when available. The resulting representation preserves the global organization of the document, including section transitions and the distribution of visually salient content, at a substantially lower visual cost than full-resolution page images.

B.2 Agent–Environment Interaction Protocol

We adopt the vanilla ReAct framework (Yao et al., 2022), which interleaves reasoning and acting within a unified interaction loop. At each step, the agent observes the question, environment state, interaction history, and available actions, and selects the next action. The environment executes the action, returns the corresponding observation, and updates its state for the subsequent decision.

Action Space. DocPilot exposes six actions: `<Analyze>`, `<Navigate>`, `<Extract>`, `<Focus>`, `<Solve>`, and `<Answer>`. `<Analyze>` decomposes the question, updates the solving plan, assesses evidence sufficiency, diagnoses failures, and verifies candidate answers. `<Navigate>` inspects page thumbnails or document overviews to identify potentially relevant pages. `<Extract>` selects task-relevant evidence from OCR-derived text, tables, visual elements, and page images. `<Focus>` performs fine-grained visual inspection through cropping, zooming, or subfigure isolation. `<Solve>` generates executable Python code over the collected evidence, which the environment executes to return the results. `<Answer>` synthesizes a candidate response from the accumulated evidence.

State-conditioned Action Availability. The available action set is dynamically determined by the current environment state, which records navigation progress, candidate and processed pages, and candidate answers with their verification outcomes. These states jointly determine which actions are valid at each step. For example, `<Extract>` requires selected pages, `<Focus>` requires available visual evidence, and `<Solve>` requires extracted evidence for computation. When the next step is ambiguous or evidence is insufficient, the agent may invoke `<Analyze>` to formulate or revise its plan; otherwise, it can directly execute the appropriate action without redundant analysis. This state-conditioned interface prevents invalid operations while allowing flexible composition of different capabilities.

Execution and State Update. Before execution, the environment validates the selected action, its arguments, and referenced page or evidence identifiers. Invalid actions or references leave the environment unchanged. After successful execution, the resulting observation is appended to the interaction history, while relevant states, including candidate pages, extracted evidence, focused regions, computation results, candidate answers, and verification feedback, are updated accordingly. This persistent state allows subsequent decisions to build directly on previous interactions.

Verification and Termination. `<Answer>` produces a candidate response, which can be verified by `<Analyze>` against the question and accumulated evidence. If accepted, the interaction terminates; otherwise, identified evidence gaps or inconsistencies guide further evidence seeking, reasoning, computation, or answer revision. The interaction also stops at the maximum step budget, returning "INSUFFICIENT_EVIDENCE" if no sufficiently supported answer is obtained.

Table 6: Datasets used for data-grounded trajectory synthesis and their capability coverage.

Type		Source Datasets	Primary Capabilities
Single-ability	<Navigate>	DocSeeker (Yan et al., 2026b)	Page-level navigation
	<Extract>	HotpotQA (Yang et al., 2018)	Evidence localization
	<Solve>	File-CSV (Zhang et al., 2025b)	Code Generation
	<Answer>	FinQA (Chen et al., 2021)	Table reasoning
	<Answer>	ChartQAPro (Masry et al., 2025), InfographicVQA (Mathew et al., 2022)	Chart and visual reasoning
Multi-ability		LongDocURL (Deng et al., 2025), DUDE (Van Landeghem et al., 2023) MADQA (Borchmann et al., 2026), MMDocRAG (Dong et al., 2026)	Long-document, multimodal, and cross-document reasoning

B.3 Module Design Details

This appendix exposes four specialized tools with explicit descriptions and structured argument schemas. <Navigate> scans page thumbnails to locate potentially relevant pages, while <Extract> retrieves question-relevant evidence from the selected pages. When finer visual interpretation is required, <Focus> applies targeted operations such as zooming, cropping, and subfigure isolation. For numerical or symbolic reasoning, <Solve> generates and executes Python code over the collected evidence. At each step, the agent conditions on the tool descriptions, current evidence state, and interaction history to determine which tool to invoke and how to instantiate its arguments, enabling adaptive tool selection without a fixed execution sequence. Detailed prompts and implementation specifications for each tool are provided in Appendix F.

Tool schema

<Navigate>. Inspect page thumbnails to identify relevant pages and return candidates with reasons.

Field	Type	Description
query	string	Navigation goal or analyzed question.
doc_id	string	Document ID.
pages	string	Optional pages or ranges to inspect.
force_refresh	boolean	Reinspect previously verified pages if enabled.
refresh_reason	string	Reason for refreshing previously verified pages.

<Extract>. Extract question-relevant evidence from selected pages.

Field	Type	Description
pages	string	Candidate pages returned by navigation
instruction	string	Task-specific extraction goal.
max_pages	integer	Maximum number of pages to process.

<Focus>. Perform fine-grained visual inspection on extracted evidence.

Field	Type	Description
instruction	string	Visual inspection goal.
figure_refs	array	Figure references from extraction.
operations	array	Visual operations for the target figure, e.g., zoom, crop, or subfigure isolation.

<Solve>. Generate Python code for deterministic reasoning over collected evidence.

Field	Type	Description
instruction	string	Code-generation or computation goal.
evidence_ids	array	Evidence items used for computation.

C Trajectory Synthesis Details

Motivation. Existing DocVQA datasets predominantly provide document–question–answer triplets, without supervising the intermediate decisions required for interactive document solving. This supervision is insufficient for learning both the execution of individual tools and their adaptive composition over long trajectories. Collecting such traces manually is also costly, as it requires repeated document inspection, evidence annotation, tool execution, and answer verification. We therefore construct DocAgent-Instruct-40K through data-grounded trajectory synthesis, using

Table 7: Statistics of DOCAGENT-INSTRUCT-40K.

Trace Type	Capability	Number
Single-ability	<Navigate>	13,986
	<Answer>	10,749
	<Extract>	6,196
	<Solve>	3,007
	Subtotal	33,938
Multi-ability	Interaction traces	4,752
Total		**38,690**

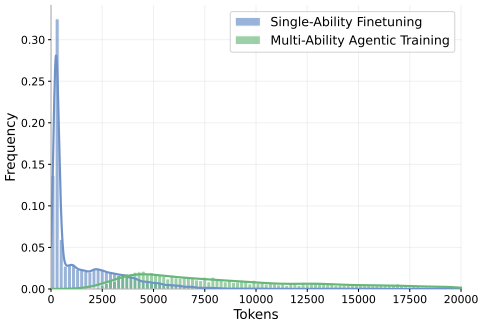

Figure 8: Length distribution of training data.

existing annotations to constrain teacher generation and ensure that the resulting traces remain executable and evidence-grounded.

Data Sources. As shown in Table 6, we select datasets according to the capabilities required by DOCPILOT. For single-ability supervision, DocSeeker provides page-level evidence annotations for <Navigate>, while HotpotQA supplies multi-hop evidence-seeking examples for <Extract>. For <Solve>, we use the `file_csv` subset of DeepAnalyze, where each instance pairs a table and question with target code, providing supervision for code-based reasoning. For <Answer>, we synthesize reasoning traces from QA datasets spanning multiple modalities: FinQA for table reasoning, ChartQAPro for chart reasoning, and InfographicVQA for visually rich document understanding. These datasets are used to synthesize focused reasoning traces for individual capabilities.

For multi-ability supervision, we use existing DocVQA benchmarks that require longer and more heterogeneous interaction. LongDocURL contains long-document questions with dispersed evidence; DUDE covers diverse document layouts and question types; MADQA emphasizes cross-page and cross-document reasoning; and MMDocRAG requires retrieval and reasoning over multimodal document collections. Together, these datasets provide realistic tasks for synthesizing complete interaction trajectories involving evidence seeking, multimodal reasoning, and verification.

Teacher-Guided Synthesis. We use Qwen3.5-397B-A17B as our multimodal synthesis teacher.

For *reasoning trace synthesis*, we construct capability-specific prompts from the question, task input, and target output. The teacher generates focused traces for locating evidence-bearing pages, extracting grounded evidence, answering questions across document modalities, or producing executable Python code. Generated code is executed to validate its result, while other traces are checked against the reference answers or evidence annotations.

For *interaction trace synthesis*, the teacher is rolled out within the full DOCPILOT environment. At step t, it selects an action and its arguments conditioned on the question q, interaction history H_t, and accumulated evidence E_t:

$$(a_t, \phi_t) \sim \mathcal{T}(q, H_t, E_t)\,, \tag{5}$$

where $\mathcal{T}$ denotes the teacher model. The environment executes the action and returns observation o_t, after which the evidence state and interaction history are updated:

$$E_{t+1} = \mathrm{Update}(E_t, o_t), \qquad H_{t+1} = H_t \cup \{(a_t, \phi_t, o_t)\}. \tag{6}$$

This process continues until the teacher produces a verified answer or reaches the interaction budget. No fixed action sequence is imposed: the teacher may revisit navigation, extraction, visual inspection, or computation as new evidence gaps emerge.

We apply three quality-control criteria to the resulting trajectories. First, *answer correctness* requires the final prediction to match the reference answer under the corresponding dataset metric. Second, *interaction validity* checks tool names, argument schemas, document references, and executable code. Third, *evidence grounding* verifies that the answer is supported by the evidence collected during interaction. We further discard malformed trajectories, repeated action loops, and executions containing no useful evidence acquisition.

Data Format and Statistics. Table 7 summarizes the dataset composition, while Fig. 8 presents the token-length distributions across the two training stages. After filtering, DOCAGENT-INSTRUCT-40K contains approximately 40K examples: 33,938 single-ability reasoning traces for developing

Table 8: Average number of tool calls under different training strategies on MMLongBench-Doc.

Training Methods	Navigate	Extract	Focus	Solve	Analyze	Answer	Total
Curriculum-based Agentic Training	2.87	2.07	1.46	0.85	8.20	0.95	16.41
-Only Multi-ability Fine-tuning	3.75	2.03	0.74	0.71	8.21	0.98	16.43

individual document-solving capabilities and 4,752 multi-ability interaction traces for learning their adaptive composition over complete trajectories. We set the maximum sequence lengths to 8K and 20K text tokens for the two stages, respectively, excluding visual tokens.

D More Evaluation Results

D.1 Evidence Recall across Navigation Rounds

We examine how DocPilot accumulates page-level evidence over successive navigation rounds on MMLongBench-Doc (Ma et al., 2024). Each <Navigate> action constitutes one round. For question i, let $P_{i,t}$ denote the pages retrieved at round t and G_i its ground-truth evidence pages. We measure macro-average cumulative recall as

$$R_t = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \frac{\left| \left(\bigcup_{s=1}^{t} P_{i,s} \right) \cap G_i \right|}{|G_i|}, \tag{7}$$

where $\mathcal{I}$ contains questions with nonempty evidence annotations.

Progressive evidence acquisition. Across backbones, first-round recall ranges from 45.67% to 50.71%, leaving substantial evidence undiscovered. By the third round, recall reaches 80.29%–88.06%, gaining 33.34–41.04 percentage points over the first round. Final recall rises to 90.04%–94.63%. These gains show that evidence acquisition extends substantially beyond the initial navigation decision, supporting adaptive interaction for broader evidence coverage.

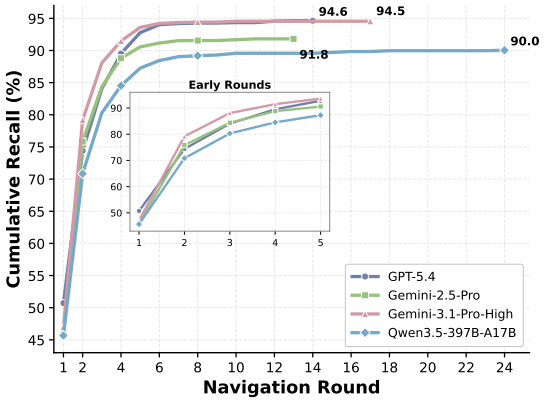

Figure 9: Curriculum-based training.

Diminishing returns. Evidence coverage expands rapidly in the early rounds, with all backbones reaching 96.9%–99.0% of their final recall by round five. Navigation beyond round six adds only 0.34–1.60 percentage points, indicating diminishing returns at greater navigation depths. Despite this aggregate saturation, individual questions may still require further evidence seeking.

Backbone-dependent convergence. Gemini-3.1-Pro achieves higher recall than GPT-5.4 from rounds two through six, while both converge to similar final coverage (94.55% versus 94.63%). Qwen3.5-397B-A17B improves more gradually, reaching 90.04% by round 24. Thus, navigation depth alone does not determine evidence coverage: backbones differ in both how quickly they acquire relevant evidence and the coverage they ultimately achieve.

D.2 Tool Usage Analysis of DocPilot-4B

Table 8 compares the tool usage of curriculum-based agentic training with direct multi-ability fine-tuning. The two strategies yield nearly identical total call counts (16.41 vs. 16.43), while <Extract>, <Analyze>, and <Answer> usage remains largely unchanged. The main difference therefore lies in how interactions are allocated across specialized capabilities.

Curriculum training reduces <Navigate> calls from 3.75 to 2.87, nearly doubles <Focus> calls from 0.74 to 1.46, and increases <Solve> calls from 0.71 to 0.85. This redistribution indicates a shift from broad page exploration toward local visual inspection and code-based reasoning over collected

Table 9: Performance comparison on MMLongBench-Doc by document type. **Bold** indicates the best result, and underlined indicates the second best. Parenthesized values for DOCPILOT-4B denote percentage-point changes over vanilla Qwen3.5-4B; green indicates gains and red indicates declines.

Model	Document Type & Domain							Overall
	Research	Acad. Paper	Guidebook	Tutorial	Finance	Brochure	Industry	
Qwen3.5-4B	54.6	52.0	54.6	62.7	51.3	52.2	59.3	54.9
Qwen3.5-9B	55.7	49.7	59.7	61.6	44.4	60.0	62.4	55.6
Qwen3.5-397B-A13B	61.9	58.8	56.3	65.3	68.8	62.0	65.4	62.0
GPT-5.4	58.3	53.0	56.1	64.5	71.8	62.0	64.2	60.1
Qwen3.7-Plus	60.5	58.8	60.5	65.9	72.6	64.9	61.1	62.6
Gemini-3.1-Pro	63.6	62.1	58.1	67.4	74.4	67.0	67.9	64.8
DOCPILOT (Ours)								
w/ Qwen3.5-397B-A13B	**75.5**	65.8	65.2	76.1	**79.5**	58.3	80.2	72.1
w/ Gemini-2.5-Pro	69.4	**76.5**	74.8	75.4	78.3	61.6	73.8	72.8
w/ Gemini-3.1-Pro	73.8	75.8	**77.6**	79.0	76.1	**68.8**	79.6	**75.6**
w/ GPT-5.4	72.7	73.2	68.8	**79.9**	74.4	67.0	**82.7**	73.5
DOCPILOT-4B	61.0$_{(+6.4)}$	57.3$_{(+5.3)}$	53.5$_{(-1.1)}$	63.8$_{(+1.1)}$	58.1$_{(+6.8)}$	53.0$_{(+0.8)}$	69.1$_{(+9.8)}$	59.1$_{(+4.2)}$

evidence. At a comparable number of interactions, the curriculum-trained agent devotes more steps to examining and reasoning over candidate evidence.

These patterns are consistent with the curriculum's progression from individual capability acquisition to adaptive orchestration. Single-ability fine-tuning provides a foundation for subsequent tool composition, potentially helping the agent move beyond repeated navigation toward finer-grained evidence processing. This offers a plausible explanation for the curriculum's performance gains, although call frequencies alone do not establish whether each invocation is appropriate or successful.

D.3 PERFORMANCE ACROSS DOCUMENT TYPES.

Table 9 evaluates DOCPILOT across seven document types on MMLongBench-Doc. DOCPILOT achieves the best results in every category, improving overall accuracy by 10.1–13.4 percentage points over the corresponding backbones. Substantial gains span multiple document types: GPT-5.4 improves by 20.2 points on academic papers and 18.5 points on industry documents, while Gemini-3.1-Pro gains 19.5 points on guidebooks. These improvements demonstrate that the benefits of interactive document solving extend beyond a single domain.

The strongest backbone nevertheless varies by document type. Gemini-3.1-Pro achieves the highest overall accuracy (75.6), while GPT-5.4 leads on tutorials and industry documents, and Qwen3.5-397B leads on research and finance. Gains are also less consistent on brochures: Gemini-3.1-Pro and GPT-5.4 improve by 1.8 and 5.0 points, respectively, whereas Qwen3.5-397B declines from 62.0 to 58.3. This variation highlights the broad applicability of DOCPILOT while identifying document-specific limitations that remain sensitive to the underlying backbone.

DOCPILOT-4B improves overall accuracy from 54.9 to 59.1, with gains across six of seven document types. The largest improvements occur on industry (+9.8), finance (+6.8), and research (+6.4), extending the benefits of interactive document solving to compact models.

E CASE STUDY

We examine three representative cases of DOCPILOT-4B after curriculum-based training, covering numerical computation, fine-grained visual understanding, and cross-page semantic comparison. These trajectories illustrate how the trained agent composes individual capabilities according to the question and intermediate observations.

Numerical Computation. In Fig. 10, the question requires combining dataset statistics with a distribution chart to determine a claim count. DOCPILOT-4B uses `<Navigate>` and `<Extract>` to locate the relevant table and figure, then invokes `<Focus>` to inspect the chart before proceeding to `<Solve>`. The resulting numerical reasoning yields a response of 245, whereas vanilla Qwen3.5-4B returns `unknown`. This case illustrates the composition of evidence seeking, visual interpretation, and

numerical reasoning: the agent first resolves the required quantities, then uses them to formulate the calculation.

Fine-Grained Visual Understanding. Fig. 11 asks when a blue bar begins in a timer chart. After locating the specified page and extracting the figure, DOCPILOT-4B performs two successive <Focus> operations to inspect the bar and its time label. It correctly answers 15:00, distinguishing the visual evidence from a nearby textual example mentioning 21:00, which matches the vanilla model's incorrect response. The case highlights the importance of adjusting inspection granularity: page-level relevance alone does not resolve a question whose answer depends on a small visual detail.

Cross-Page Semantic Comparison. In Fig. 12, the agent must count figures conveying a message similar to the first figure. DOCPILOT-4B establishes the reference message, then repeatedly invokes <Navigate> and <Extract> to collect comparison evidence across the main paper and appendices. Following an unsuccessful <Solve> attempt, <Analyze> redirects the process to <Answer> for semantic comparison, ultimately selecting Figures 4 and 5. This trajectory illustrates sustained evidence acquisition and adjustment after tool failure, extending orchestration beyond a fixed sequence of successful actions.

Cross-Document Reasoning. In Fig. 13, DOCPILOT-4B starts from document overviews and invokes <Navigate> and <Extract> to gather evidence from the FEC settlement and TN Board documents. <Analyze> identifies mismatches between the retrieved titles and the requested role, prompting re-navigation to the TN Board's signature page. Linking the settlement year, 2023, to the official's title and signature date yields Alex Martin. This trajectory illustrates how intermediate observations guide document switching and role disambiguation to support cross-document evidence integration.

These distinct interaction patterns qualitatively support the curriculum's goal of connecting individual document-solving capabilities with adaptive orchestration.

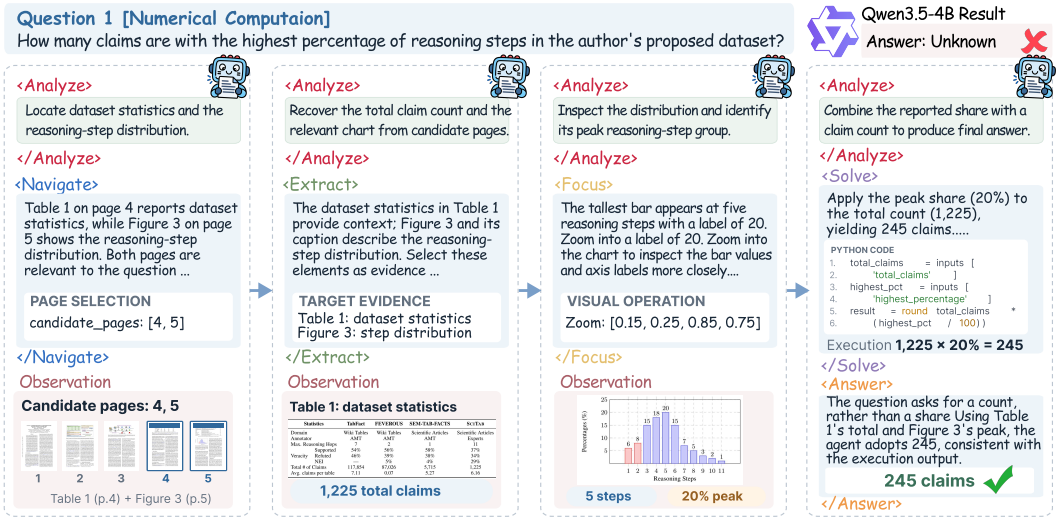

Figure 10: A numerical computation case study showing DOCPILOT-4B's orchestration.

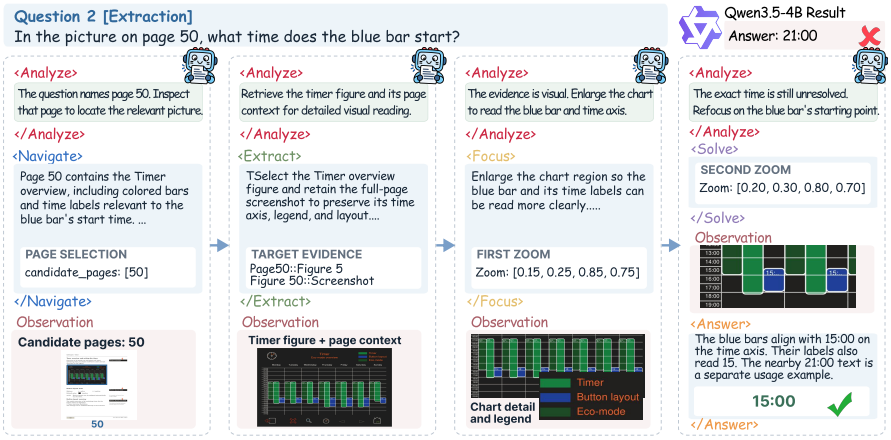

Figure 11: An evidence extraction case study showing DocPilot-4B's orchestration.

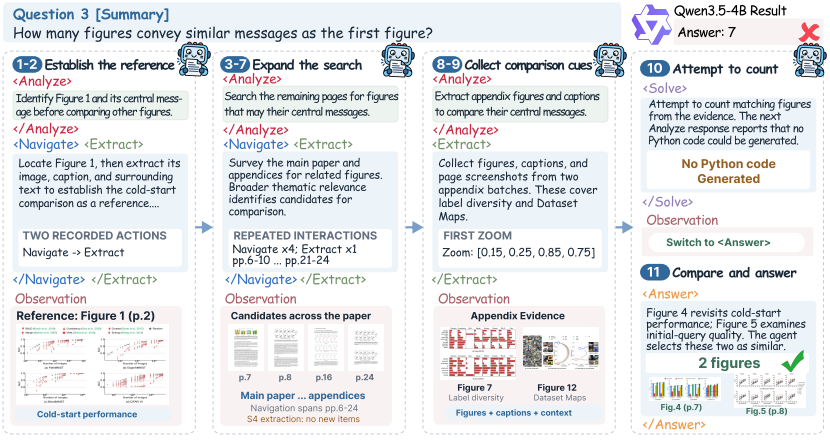

Figure 12: A counting summary case study showing DocPilot-4B's orchestration.



Figure 13: A cross-document reasoning case study showing DocPilot-4B's orchestration.

F SYSTEM PROMPT DESIGN

Navigate Agent Prompt

[System Role & Objective]
You are the Navigate Agent in a Document Visual Question Answering (DocVQA) system. Your objective is to scan document pages and identify **ANY** page with potential to contribute to answering the search query.

[Crucial Constraint]
You are **EXCLUSIVELY** a page-level navigator.
Do **NOT** attempt to answer the user's question.
Do **NOT** suggest downstream tools, extraction strategies, or region-level focus operations.

[Page Selection & Navigation Strategy]
1. High Recall for NEW Pages: Include any new page containing exact keywords, synonyms, related thematic concepts, or structural elements such as tables or figures that might hold the answer. Do not discard new pages using a rigid confidence threshold; use the `confidence` score to rank relevance.
2. High Precision for PREVIOUS Candidates: Re-evaluate previously identified candidate pages rigorously and retain only those with strong ongoing relevance.
3. Avoid Redundant Work: Do not re-select pages that have already been fully verified by current evidence.
4. Multi-Hop Look-Ahead: Set `need_more_pages=true` when current evidence is incomplete or unvisited pages may contain missing evidence for multi-step reasoning.
5. Empty Selection: If no current page is relevant and no previous candidate remains strongly relevant, return an empty `candidate_pages` array and explain why in `reason`.
6. No Out-of-Range Pages: Only select page numbers appearing in the input `page_set` or `pages`. Never invent or extrapolate page numbers.
7. Unavailable Target Pages: If the query requests a specific page that is absent from the current input, return `candidate_pages: []`, explain that the page is unavailable, and do not substitute nearby pages unless explicitly relevant to the same evidence need.

[Strict Output Constraints]
Page Format: Use integer page numbers only.
Page Boundary: Every selected page must appear in the input `page_set`.
Output Schema: Return only a valid JSON object matching the format below,

```
{
  "candidate_pages": [
    {
      "page": 1,
      "confidence": 0.85
    }
  ],
  "reason": "brief explanation of why the selected pages are relevant",
  "need_more_pages": true
}
```

Extract Agent Prompt

[System Role & Objective]
You are the Extract Agent in a Document Visual Question Answering (DocVQA) system.
Your objective is to evaluate candidate items from selected pages and retain only those containing specific, actionable evidence required to answer the search query.

[Input Context]
You will receive the **search query** and a **candidate list** containing natural-language descriptions of Text, Table, Figure, and page screenshot nodes.

[Core Selection Strategy]
1. Answerability is Key: Select only items containing the facts, values, qualifiers, labels, definitions, or visual/structural evidence required to resolve the query. Reject generic or tangential content.
2. Evidence Completeness: Ensure the selected evidence covers all required answer slots, including entity or category, metric, time period, units, labels, and calculation inputs. If necessary, select the smallest combination of items that provides complete support.
3. Precision with Coverage: Keep the selection focused while retaining evidence required to interpret the

answer. If local nodes may omit important layout, legends, paired figures, or cross-node relationships, also select the full-page screenshot as supporting visual evidence.
4. Mixed Evidence: Select evidence across modalities when the query requires cross-referencing text, tables, figures, captions, or legends. Do not assume numerical questions can be resolved from tables alone when surrounding text provides necessary context or constraints.
5. Empty Return: If none of the candidates provide useful evidence, return an empty list.

[Modality-Specific Prioritization]
Visual & Spatial Queries: Prioritize relevant Figure nodes or page screenshots. For multi-panel figures, select the containing Figure node for downstream subfigure inspection.
Textual Queries: Prioritize relevant Text nodes with definitions, context, quotes, or facts.
Tabular Queries: Prioritize relevant Table nodes for lookups, comparisons, totals, or rankings.
Mixed Queries: Select evidence across modalities when jointly required.

[Strict Output Constraints]
Return only a valid JSON object matching the schema below.
Do **NOT** include Markdown formatting or any text outside the JSON object.

```
{
  "selected_candidate_ids": [
    "page7::text::7",
    "page7::screenshot"
  ],
 "reason": "A brief, one-sentence explanation of how the selected items support the query."
}
```

Focus Agent Prompt

[System Role & Objective]
You are the Focus Planner in a Document Visual Question Answering (DocVQA) system.
Your objective is to plan high-precision visual reading operations for an already extracted figure.

[Crucial Constraint]
Output only visual operations. Do **NOT** attempt to answer the user's question directly.

[Allowed Operations]
- `subfigure_split`: Use for multi-panel figures. Specify the target panel in `region.panel_query` and provide `grid_hint` when the layout is clear. Do not combine it with full-figure crop or zoom.
- `crop`: Use for clearly bounded regions such as source notes, footer text, or distinct textual blocks.
- `zoom`: Use for small labels, legends, axis labels, or exact numerical values.

[Region Specification]
Specify the target region in the `region` object. Prefer normalized coordinates `[x1, y1, x2, y2]` in the $[0, 1]$ figure-relative space when the region is geometrically clear; otherwise, use a descriptive coarse region.

[Visual Captioning]
The `visual_caption` is required and should briefly summarize the localized visual content and relevant entities, labels, values, trends, panel identities, or legend mappings.
Do **NOT** state the final answer unless it is directly visible in the selected region, and avoid instructional descriptions such as "inspect labels" or "zoom into the chart."

[Output Format]
Return only valid JSON following the schema of the selected operation. Use the minimum number of visual operations required for the current focus goal, and do not combine incompatible operations.

Solve Agent Prompt

[System Role & Objective]
You are the Solver Agent in a Document Visual Question Answering (DocVQA) system.
Your objective is to generate robust Python code for deterministic computation over extracted structured evidence, especially tabular data.

[Execution Mechanism]
The generated `code` is executed by the environment using variables provided in `inputs`.
The final computed answer must be assigned to the Python variable `result`.

[Coding Rules]
1. Variable Requirement: Always assign the final output to `result`.
2. Input Dependency: Read data only from the `inputs` dictionary.
3. Allowed Imports: Only `pandas`, `numpy`, `re`, and `math` may be imported. Do not access files, APIs, or the network.
4. Determinism: Keep all computation deterministic and reproducible.

[Data Processing Standards]
Data Cleaning: Normalize formatted values such as currencies, percentages, missing values, and accounting-style negatives before computation.
DataFrame Processing: For tabular evidence, store records in `inputs` and convert them to a Pandas DataFrame for filtering, aggregation, or comparison.
Error Handling: Guard against empty results, invalid values, and division by zero.
Reported Quantity: Use the explicitly reported values requested by the question. Do not reconstruct alternative quantities unless the target value is absent and the evidence explicitly defines the derivation.

[Fallback]
If the evidence is insufficient for grounded computation, return:

```
{
"summary": "Evidence is insufficient, so the solver returns result = None.",
"code": "result = None",
"inputs": {}
}
```

[Strict Output Constraints]
Return only a valid JSON object containing `summary`, `code`, and `inputs`.
Do not include Markdown formatting or additional text. Python newlines in `code` must be escaped as `\n`.

Answer Agent Prompt

[System Role & Objective]
You are the unified Answer Agent in a Document Visual Question Answering (DocVQA) pipeline.
Your objective is to generate a concise, factual answer based **EXCLUSIVELY** on the provided evidence.

[Grounding & Anti-Hallucination]
Strict Grounding: Rely entirely on the provided evidence. Do not invent values, entities, relationships, or assumptions.
Page Grounding: Mention only page numbers explicitly present in the evidence.

[Reasoning & Deduction]
Rigorous Reasoning: Apply the question criteria systematically and connect all required evidence into a complete reasoning chain.
Semantic Alignment: Match the target entity, metric, time period, unit, and comparison scope to the evidence. Return "Not Answerable" when explicit constraints conflict.
Use the Reported Quantity: Use values reported under the labels named in the question. Reconstruct quantities only when the requested value is absent and the document explicitly defines the derivation.
Count vs. Share: Return percentages only when explicitly requested. For headcount questions, compute from the relevant sample size and rate; return "INSUFFICIENT_EVIDENCE" if the required sample size is unavailable.
Consistency: The final answer must follow directly from the reasoning in the rationale.

[Fallback Rules]
Deterministic Derivation: When all required numerical inputs are available, compute the requested value directly rather than abstaining.
Document-provided Fallback: Use an explicit default, remaining, or fallback value when the requested item is not separately listed but the document provides an applicable alternative.
Incomplete Evidence: Return "INSUFFICIENT_EVIDENCE" only when a required fact is missing and no deterministic derivation or document-provided fallback is available.
Exactness Limit: If an exact answer is requested but the evidence supports only a vague visual estimate, return "Not Answerable".

[Output Format]
Return only a valid JSON object matching the schema below,

```
{
"answer": "The concise final answer, or a fallback string (Not Answerable).",
```

```
"rationale": "A brief evidence-based explanation.",
"confidence": 0.95
}
```

Analyze Agent Prompt

[System Role & Objective]
You are the Textual Analysis Agent within the DocPilot system.
Your sole responsibility is to verify whether the current candidate answer is fully supported by the collected evidence and aligned with the original question. You reason only over the provided textual state and do not directly execute tools.

[General Rules]
Strict Grounding: Base verification exclusively on the provided question, evidence, and current state. Do not introduce external knowledge.
Answer Type: Infer the expected answer type from the question, such as entity, percentage, ratio, or count, and verify that the candidate matches it.
Page Grounding: Mention only page numbers explicitly present in the supporting evidence.
Support vs. Coverage: Distinguish whether the candidate is supported by current evidence from whether the search coverage is sufficient. Acceptance requires both.

[Failure Diagnosis & Tool Mapping]
`navigate`: Use when relevant pages may remain unexplored or page-level coverage is incomplete.
`extract`: Use when the correct page is known but required text, table, figure, or related nodes are missing.
`focus`: Use when visual evidence is available but labels, subfigures, or spatial details remain unresolved.
`solve`: Use when evidence is available but deterministic computation, or comparison remains unresolved.
`answer`: Use when evidence is sufficient but the candidate answer is poorly synthesized or misaligned.

[Verification Criteria]
Accept the candidate when it is grounded, internally consistent, and covers the requested scope.
Reject only for clear fatal flaws, such as inconsistent reasoning, unsupported conclusions, incomplete comparisons, incorrect counts, or missing required evidence.
Do not reject based on speculation, unused contextual facts, or generic uncertainty.

[Unanswerable Cases]
`Not Answerable` may be accepted when the evidence sufficiently establishes that the requested information is unavailable, outside the document scope, or not applicable.

[Output Format]
Return only a valid JSON object matching the schema below,

```
{
"result": true,
"issues": "Empty when accepted; otherwise state the decisive flaw or evidence gap.",
"suggestion": "Empty when accepted; otherwise list next action",
"summary": "Briefly summarize the verification result."
}
```