# MINERU-POPO: UNIVERSAL POST-PROCESSING MODEL FOR STRUCTURED DOCUMENT PARSING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

VLM-based OCR models have become the de facto choice for document parsing, as they can accurately extract page-level elements (e.g., paragraphs within individual pages) together with their bounding boxes and textual content. However, downstream applications such as RAG require coherent document-level information, whereas these models often break cross-page continuity and fail to recover disrupted structures, such as paragraphs and tables truncated by page boundaries. Such relationships are not confined to a single page; instead, they require joint analysis of titles, paragraphs, tables, and images spanning multiple pages. A natural solution is therefore to reuse existing OCR outputs and reconstruct document-level logical structures through post-processing.

To this end, we propose MinerU-Popo, a lightweight and universal framework for **PO**st-**P**rocessing **O**CR outputs, which converts page-level results from diverse parsers into coherent document-level structures. MinerU-Popo decomposes the problem into four focused subtasks: text truncation recovery, table truncation recovery, title hierarchy reconstruction, and image-text association. To address these effectively, we build a task-oriented data engine with task-specific input filtering, and use the generated data (**30K**) to fine-tune a lightweight post-processing model (Qwen3-VL-4B). To support long documents, we introduce dynamic chunking with overlap-based synchronization, which aligns chunk-level outputs from the fine-tuned model and preserves global consistency. Finally, we assemble the aligned outputs into a tree-structured document representation, further enriched with node chunking and summaries for downstream retrieval and analysis. Empirical results show MinerU-Popo improves title-hierarchy TEDS by at least 20% across all five tested OCR models, improves RAG accuracy on most subsets, and reduces per-query latency by up to 70% compared with raw OCR baselines.

## 1 INTRODUCTION

Document parsing converts raw documents (e.g., in PDF or image formats) into machine-readable representations that support downstream retrieval, question answering, and automated analysis (Xu et al., 2026; Lin et al., 2024; Sun et al., 2025; Saad-Falcon et al., 2024). Recent OCR models, such as MinerU (Wang et al., 2026), PaddleOCR (Cui et al., 2026), GLM-OCR (Duan et al., 2026), Dolphin (Feng et al., 2026) and MonkeyOCR (Zhang et al., 2025a), have substantially improved page-level parsing by extracting element types, bounding boxes, reading order, and textual content from individual pages. Moreover, although general VLMs are powerful, they remain costly and intensive at scale, with less controllable and standardized outputs than those of specialized OCR models. These lightweight OCR models therefore remain necessary for document parsing tasks that requires throughput and standardization (Ben Ammar, 2026). However, downstream applications (e.g., RAG) rarely operate on isolated pages. They require coherent document-level structures, while page-wise OCR often breaks cross-page continuity and obscures relationships such as heading hierarchies, figure-caption links, and paragraphs or tables truncated by page boundaries. For example, Figure 1 shows a table split across two pages, where accurate reconstruction requires jointly analyzing cross-page table continuity and the surrounding title hierarchy.

A natural solution is therefore to reuse existing OCR outputs and reconstruct document-level logical structure through post-processing. This direction is appealing because modern OCR models already



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.

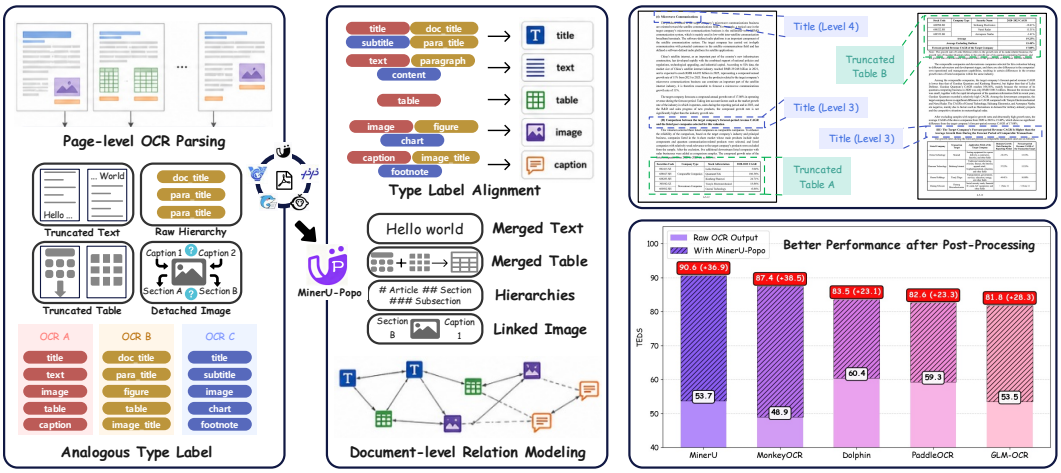



Figure 1: MinerU-Popo supports diverse OCR models through type-label alignment and bridges page-level parsing with document-level modeling via four post-processing subtasks. Applying MinerU-Popo after OCR yields clear improvements in title hierarchy analysis.

expose a broadly compatible intermediate representation: a reading-order sequence of typed blocks, each associated with content and coordinates. Moreover, these block types (e.g., text, title, image, table, caption) are largely aligned across systems and can often be normalized with lightweight rules. However, resorting to manual correction or large general-purpose VLMs for post-processing is costly for massive documents, difficult to scale, and often incompatible with privacy-sensitive or on-premise deployment (Zhang et al., 2025b; Wang et al., 2026). For instance, according to telemetry from a public document parsing platform (Wang et al., 2026), the service receives millions of OCR requests per month, with each document containing more than 20 pages on average.

However, realizing effective document post-processing is non-trivial. There are three fundamental challenges: ***(1) Cross-Page Geometric Discontinuity:*** Pagination breaks the document's logical flow, causing related elements to appear in separate pages, such as $(i)$ a section heading at the bottom of one page may govern content on the next page, and $(ii)$ a table may be split across two pages. ***(2) Redundant Document Parsing:*** OCR parsers produce comprehensive but redundant outputs that include detailed information for all element blocks. In contrast, each post-processing subtask only depends on a small subset of relevant elements, while extraneous content may introduce interference. For instance, text truncation recovery should focus on paragraph fragments interrupted by page boundaries, rather than complete paragraphs or unrelated neighboring blocks. ***(3) Scalability to Long Documents:*** Processing exceptionally long documents, such as those with hundreds of pages, in a single pass exceeds the context limits of lightweight models, while naive chunking can produce inconsistent hierarchy decisions across chunks.

To address these challenges, we introduce MinerU-Popo, a unified document post-processing framework that bridges OCR outputs and document-level structural parsing across diverse mainstream OCR models. To synthesizes page-level OCR predictions into a document-level logical structure, MinerU-Popo refines them through a locally deployed fine-tuned model that performs four document-structure subtasks: table truncation analysis, text truncation analysis, title hierarchy analysis, and image-text association analysis. To support robust structural reasoning, we construct training data that captures the diversity and representativeness of real-world complex document layouts, and further introduce task-specific data filtering to retain only the most critical evidence in the model input. To handle long documents, MinerU-Popo employs dynamic chunking and synchronization, which preserves global consistency while reducing deviations across independently processed chunks. Finally, MinerU-Popo enriches the refined parsing results through long-node chunking and summary generation, transforming fragmented OCR outputs into a unified, hierarchical, and semantically detailed document representation suitable for downstream retrieval and analysis.

Our contributions are summarized as follows:

(1) We propose MinerU-Popo, a lightweight and universal framework for post-processing OCR outputs into coherent document-level structures.

Table 1: Current OCR Models' Capability to Handle Cross-page Relationships.

| **Model** | MinerU | Paddle | GLM-OCR | Dolphin | Monkey |
|---|---|---|---|---|---|
| **Table Truncation** | Supported | Partial | Unsupported | Unsupported | Supported |
| **Text Truncation** | Partial | Partial | Partial | Unsupported | Partial |
| **Title Hierarchy** | Unsupported | Supported | Partial | Partial | Supported |
| **Image-Text Association** | Unsupported | Unsupported | Unsupported | Unsupported | Unsupported |

(2) We develop a task-oriented data engine with task-specific input filtering, enabling a lightweight model to solve four focused cross-page structural parsing subtasks efficiently.

(3) We introduce dynamic chunking with overlap-based synchronization and an enriched tree representation, improving both long-document consistency and downstream retrieval quality.

(4) Empirical results show MinerU-Popo generally improves OCR parsing result (e.g., more than 20% hierarchy TEDS improvement for 5 OCRs), improved answering performance on RAG and End-to-End QA, and up to 70% lower per-query latency of RAG.

## 2 Related Work

**OCR Models.** Recently, VLMs have driven a paradigm shift in OCR, extending beyond text recognition to comprehensive page layout reconstruction. Modern OCR models such as MinerU (Wang et al., 2026), PaddleOCR (Cui et al., 2026), GLM-OCR (Duan et al., 2026), Dolphin (Feng et al., 2026), and MonkeyOCR (Zhang et al., 2025a) have established state-of-the-art page-level parsing performance on benchmarks like OmniDocBench (Ouyang et al., 2025). However, as compared in Table 1, these models exhibit limitations in handling document-level relationships: ***(1) Text Truncation:*** Most models detect intra-page truncations but fail to recover paragraphs broken by page boundaries. ***(2) Title Hierarchy:*** GLM-OCR and Dolphin rely on rigid, predefined title levels (e.g., GLM-OCR distinguishes only two levels, `doc_title` and `para_title`, and Dolphin supports three), making them inadequate for documents with deeply nested hierarchical structures. In contrast, MinerU-Popo predicts open-ended levels to naturally accommodate arbitrary depths. ***(3) Table Truncation:*** While PaddleOCR merges truncated tables, it cannot unify adjacent cells across page boundaries. ***(4) Image-Text Association:*** None of the baselines capture semantic associations between images, captions, and sections. Ultimately, these deficiencies preclude the accurate parsing of multi-page documents.

**Hierarchy Analysis.** Hierarchy analysis models the nested structure of document elements. End-to-end approaches, such as DocHieNet (Xing et al., 2024) and HRDoc (Ma et al., 2023), infer relationships based on OCR outputs but underutilize explicit OCR priors (e.g., element types and reading order), forcing models to learn them implicitly. ZenDB (Lin et al., 2024) clusters elements using visual metadata (e.g., font size) from born-digital documents, which lacks generalizability to scanned documents and ignores semantic context. In contrast, MinerU-Popo explicitly exploits OCR-derived types and reading order to simplify model input via task-specific data filtering. By decomposing hierarchy analysis into focused subtasks (e.g., title structuring and image-text association), each operates exclusively on relevant elements, eliminating global processing overhead.

**Document QA.** Document-based question answering (QA) is a pivotal downstream task. Directly processing entire documents with VLMs is unscalable due to context-length degradation and prohibitive costs. Recent retrieval-augmented paradigms adopt two main trajectories: Vision-centric approaches (Cho et al., 2024; Chen et al., 2024) leverage image embeddings for page-level retrieval; Structure-aware frameworks (Xu et al., 2026; Lin et al., 2024; Sun et al., 2025; Saad-Falcon et al., 2024) model documents as trees or graphs for fine-grained, node-level retrieval. Post-processing provides information necessary for document modeling, such as the image-text association and title hierarchy required by ZenDB (Lin et al., 2024) and MoDora (Xu et al., 2026), liberating them from the originally simplistic rule-based approaches. Meanwhile, benchmarks like ViDoRe V3 (Loison et al., 2026) and MMDA (Xu et al., 2026) have been introduced to rigorously evaluate these systems.

## 3 Problem Formulation

The goal of post-processing is to bridge this gap by taking a sequence of OCR-extracted block elements $\mathcal{E} = \{e_1, e_2, \ldots, e_n\}$ as input, and outputting a coherent, document-level structured tree

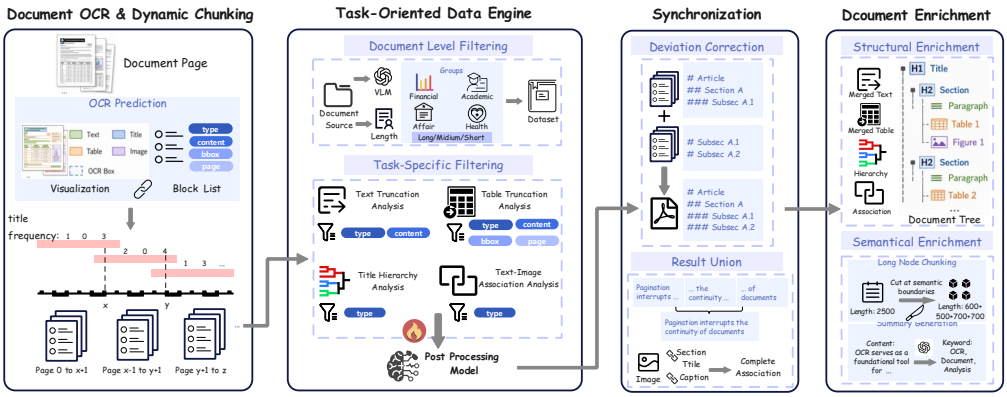



Figure 2: Overview of MinerU-Popo. The post-processing model training and inference are supported by a task-oriented data engine that collects complex and representative documents as datasets and simplifies the model input for each subtask. Page-level OCR predictions of long documents are grouped via dynamic chunking, and then synchronized after chunk-level post-processing. The result is enriched via structural tree construction as well as semantic chunking and summarization.

$\mathcal{T}$. Crucially, this process must address the challenges of cross-page discontinuity and logical structure absence, while strictly balancing ***Accuracy*** (precisely recovering truncated elements and deep hierarchies) and ***Efficiency*** (minimizing computational overhead by filtering redundant OCR information). To achieve this, we decompose the overall post-processing into four focused subtasks.

**Table Truncation Analysis.** Pagination often splits a single logical table into multiple fragments across pages. For any two adjacent table elements $e_i^{table}$ and $e_{i+1}^{table}$ across a page boundary, this subtask first needs to predict a table-level continuation label $p_i \in \{0, 1\}$ to indicate whether they belong to the same logical table. If $p_i = 1$, we need to further predict a column-wise cell merging strategy $Q_i = \{q_1, \ldots, q_x\}$, where $q_j \in \{0, 1\}$ indicates whether the boundary cells in column $j$ should remain separate (simple concatenation) or be merged into one continuous cell.

**Text Truncation Analysis.** Similarly, paragraphs are frequently interrupted by page breaks, column breaks, or inserted figures. For a sequence of text elements, this subtask needs to predict a binary label $p_i \in \{0, 1\}$ for each adjacent text pair $(e_i^{text}, e_{i+1}^{text})$ to determine whether they are logically truncated and should be concatenated into a single continuous paragraph.

**Title Hierarchy Analysis.** To reconstruct the document's structural backbone, we must determine the nesting levels of all section headings. Given a sequence of title elements extracted from the document, this subtask needs to predict an open-ended integer level for each title (i.e., $e_i^{title}.level \in \{1, 2, 3, \ldots\}$), where a smaller number represents a higher hierarchical level (e.g., Level 1 for document title, Level 2 for main sections).

**Image-Text Association Analysis.** Visual elements must be grounded in their semantic context. For each caption element $e_i^{cap}$, this subtask needs to predict its associated image or table. Furthermore, to embed these visual elements into the document tree, it predicts the corresponding section title $e_j^{title}$ that logically governs each image or table.

## 4 METHODOLOGY

As illustrated in Figure 2, MinerU-Popo presents a comprehensive pipeline to transform fragmented OCR outputs into an accurate document-level representation. The pipeline begins with a lightweight model fine-tuned on a **Task-Oriented Data Engine**, which filters out redundant OCR noise to provide focused inputs for each of the four subtasks mentioned above. To efficiently handle arbitrarily long documents without losing global context, we employ a **Dynamic Chunking and Synchronization** strategy during inference. Finally, through **Document Enrichment**, the refined elements are assembled into a hierarchical tree structure, further optimized with semantic chunking and LLM-generated summaries to facilitate downstream retrieval and analysis.

### 4.1 TASK-ORIENTED DATA ENGINE

We first provide a task-oriented data engine with document-level and task-specific input filtering.

**Document-Level Filtering.** To obtain a balanced document distribution, we first use a VLM to categorize these documents according to their topics and layout characteristics. The document topics include technology, academic, education, healthcare, affair and law. And the document layout can be categorized by single/multi column, lengths and title levels. We then sample documents from each category to achieve balanced coverage across both semantics and layouts. During the sampling process, documents with typical topics and complex structures will be given higher weights to ensure the representativeness and complexity of the dataset.

Document OCR yields multi-modal elements (text, images, tables), and simply concatenating them as input introduces extraneous noise and exacerbates long-context challenges (e.g., tables are irrelevant for determining title hierarchies). By decomposing the overall task (Section 3), each subtask only requires specific element types. Thus, we employ task-specific filtering during training and inference to retain crucial relevant elements and pages, significantly reducing input complexity.

**Task-Specific Filtering.** For each subtask, we filter the most relevant elements as input via heuristics. By retaining only the crucial elements and structural priors relevant to each specific subtask, we significantly reduce input complexity and enhance the model's focus.

*(1) Filtering for Hierarchy Analysis.* Titles play a governing role over text paragraphs. Once the hierarchical levels of titles are determined, the hierarchy of the text subordinate to each title is consequently established. Therefore, in title hierarchy analysis, only elements of type "title" are filtered out and arranged according to the reading order predicted by OCR as the input.

*(2) Filtering for Image-Text Association Analysis.* Images and tables are, on one hand, associated with their respective captions, and on the other hand, subordinate to specific sections (represented by their titles) within the document. To capture these associations, we filter elements of type title, caption, image or table, discarding the massive volume of regular body text.

*(3) Filtering for Text Truncation Analysis.* Text elements dominate document length. However, truncation inherently occurs only at the boundaries of text blocks. We leverage this structural prior by applying heuristic rules based on starting prefixes (e.g., '1.1') and terminating punctuation (e.g., periods) to discard explicitly untruncated blocks. For the remaining candidates, we extract only the first and last sentences as the final input, drastically reducing token count.

*(4) Filtering for Table Truncation Analysis.* Table truncation typically results from page breaks. To identify candidates efficiently without heavy model inference, we employ layout and structural heuristics. We select tables located at page boundaries (the last element of one page and the first of the next). We then validate these candidates by checking for continuation markers (e.g., "continued" in captions), bounding box width consistency, and identical column counts.

### 4.2 DYNAMIC CHUNKING AND SYNCHRONIZATION

After filtering relevant elements, processing long documents remains a bottleneck. For instance, a financial report spans hundreds of pages and thousands of paragraphs, making single-pass processing prohibitive. This necessitates segmenting the document into chunks. However, naive chunking leads to inconsistencies in hierarchical predictions. For example, text truncation at chunking boundaries can not be detected. Another case is that the global document title (level 1) is localized to the first chunk, and subsequent chunks lacking this context can erroneously promote section titles from level 2 to 1. To mitigate the artifacts, we propose a dynamic chunking and synchronization strategy.

**Dynamic Chunking.** To prevent information loss from boundary truncation and establish reference points for synchronization, we employ a dynamic chunking strategy with overlaps. Rather than using fixed strides, we dynamically determine chunk boundaries to enrich reference information. Specifically, within a search window defined by a preset stride $s$ and threshold $t$, we select the page $p$ in range $[b_{i-1} + s - t, b_{i-1} + s + t]$ with the highest frequency $f$ of task-specific elements $e$ (e.g., "title" for hierarchy analysis) as the boundary $b_i$. We split document pages into chunks with overlaps based on the boundaries generated through multiple iterations, that is $chunk_i = (max(0, b_i - 1), min(b_{i+1} + 1, p_{max}))$. This density-maximizing mechanism guarantees sufficient reference

elements in the overlaps, reducing contingency in the deviation calculation during subsequent synchronization.

**Chunk Synchronization.** The synchronization stage aggregates chunk-level outputs into a unified document-level prediction. Due to discrete title distributions, hierarchical predictions often diverge within overlapping regions. Using the initial chunk as an anchor, we sequentially calibrate the hierarchy levels of subsequent chunks based on their average deviations compared to the former chunk at their overlap. Let $level_{i,x}$ denotes the predicted level of title $x$ in the result of chunk $i$, the deviation is $avg(level_{i,a} - level_{i+1,a})$ for all titles in the overlapping pages between two chunks. Then we calculate the final level for all titles in the next chunk by $level_{i+1,y}^{final} = level_{i+1,y} + deviation$.

For other subtasks that do not involve such deviations, since we cover all page boundaries by overlaps, the final output of truncation or association analysis is the union of each chunk's result.

### 4.3 Document Enrichment

We perform structural enrichment to enrich both the structural and semantical document-level information via hybrid node chunking and LLM-generated summarization.

**Structural Enrichment.** Based on the predicted title hierarchies and associations, we assemble elements into a tree $\mathcal{T}$. The hierarchical levels determine the parent-child relationships of section nodes, while text paragraphs, merged tables, and associated images are attached as child elements to their respective governing section nodes.

**Semantical Enrichment.** While the derived primary tree captures document structure, it remains suboptimal for downstream retrieval and analysis. First, title-based partitioning yields excessively long nodes for documents with sparse titles (e.g., novels or prose), hindering effective processing. Second, full-content embedding incurs much overhead, yet isolated titles (e.g., "Methodology") lack sufficient granularity for accurate retrieval. To address these limitations, we introduce node chunking and summary generation. During node chunking, we sequentially process each node's text. Once the accumulated text length reaches the threshold, we segment it into a sub-node at the next paragraph boundary. These boundaries are derived from OCR outputs, but truncation boundaries $(e_i^{text}, e_{i+1}^{text})$, $p_i = 1$ are excluded. Following node chunking, we prompt an LLM to generate summaries for all nodes. The resulting document tree balances optimal granularity with information completeness. Serving as an intermediate representation between titles and full texts, these summaries distill crucial arguments, methods, trends, or conclusions while abstracting away details.

## 5 Experiments

### 5.1 PostDocBench

Existing OCR benchmarks like OmniDocBench (Ouyang et al., 2025), DocLayNet (Pfitzmann et al., 2022) and olmOCR-Bench (Poznanski et al., 2025) are not suitable for our evaluation because they mainly target page-level parsing, whereas our focus is document-level post-processing. With the task-oriented data engine introduced in Section 4.1, we construct PostDocBench to evaluate post-processing performance (see Appendix A for detials). Besides, we adopt ViDoRe V3 (Loison et al., 2026) for Retrieval-Augmented Generation (RAG) evaluation, utilizing its bounding box annotations to evaluate spatial retrieval accuracy. Additionally, we use MMDA (Xu et al., 2026) for end-to-end QA evaluation to assess the representational capacity of different document structures. In all experiments, we utilize MinerU (Wang et al., 2026) as the base OCR model, fine-tune Qwen3-VL-4B (Team, 2025) for post-processing, and leverage Gemini-3-Flash (Pichai et al., 2025) for answer generation.

### 5.2 Experiment Setup

First, we support multiple mainstream OCR models, including MinerU2.5-pro (Wang et al., 2026), PaddleOCR-VL-1.5 (Cui et al., 2026), GLM-OCR (Duan et al., 2026), Dolphin-v2 (Feng et al., 2026), and MonkeyOCR v1.5 (Zhang et al., 2025a). Besides, we fine-tune Qwen3-VL-4B (Team, 2025) for post-processing, utilize Qwen3-Embedding-4B (Zhang et al., 2025c) for embedding-based retrieval, and leverage Gemini-3-Flash (Pichai et al., 2025) for final answer generation. The training is conducted on 8 H200 GPUs for 6 hours.

Table 2: Model Performances Comparison on PostDocBench.

| **Model** | MinerU-Popo | Qwen3-VL-2B | Qwen3-VL-4B | Qwen3-VL-8B | Qwen3-VL-32B |
|---|---|---|---|---|---|
| **TEDS↑** | **90.6** | 21.2 | 56.5 | 65.9 | 78.0 |
| **Doc/s↑** | **0.37** | 0.22 | 0.20 | 0.16 | 0.04 |

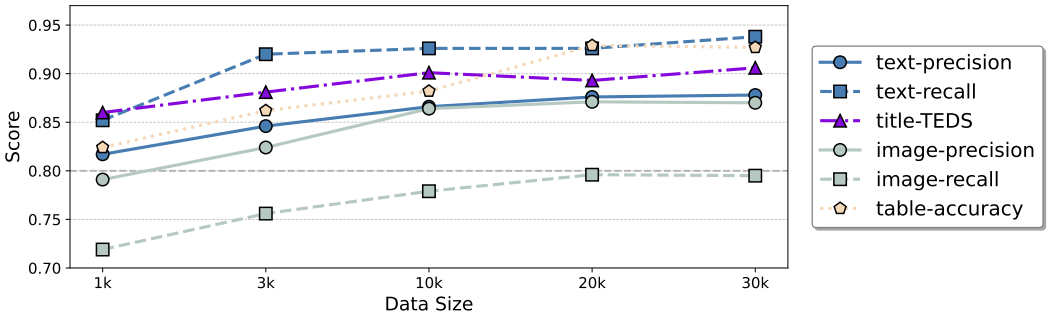



Figure 3: Performance on PostDocBench with Different Training Data Sizes.

### 5.3 Experimental Results

**Post Processing Performance.** Table 2 compares the TEDS score and inference speed, measured in documents per second, of MinerU-Popo and other pre-trained models on title hierarchy analysis under the same deployment conditions. TEDS stands for Tree-Edit-Distance-Similarity. Since the document elements are organized into a tree structure through post-processing, this metric directly reflects the correctness of the reconstructed hierarchical structure by measuring the edit distance of edges and nodes between the predicted and ground-truth document trees. MinerU-Popo achieves both a higher TEDS score than larger models, 90.6 vs. 78.0, and higher speed than even smaller models, 0.37 vs. 0.22. This shows that our task-oriented data engine and dynamic chunking improve both model training and inference efficiency by reducing input and output length.

We further evaluate the efficacy of fine-tuning by jointly training on varying subsets (1k to 30k instances per subtask) from PostDocBench. As Figure 3 illustrates, when training instances for each subtask increases from 1k to 10k, all the corresponding scores improve. But as the data size expands from 10k to 30k, the performance gain diminishes substantially. This indicates that the current training data are sufficient, and more training data would yield only marginal benefits.

For MinerU-Popo fine-tuned on the complete training set, the precision and recall for text truncation analysis reach 87.8% and 93.8%, those for image-text associations analysis are 87.0% and 79.5% respectively, while the scores of other two subtasks exceed 90% (e.g., the TEDS between predicted and annotated title hierarchy is 90.6%, the accuracy for merge method prediction in table truncation analysis is 92.7%). This demonstrates that the fine-tuned model can make accurate judgments across various post-processing subtasks, allowing us to integrate its inference results to obtain a well-formed document modeling structure.

**Finding 1:** Supported by its task-oriented data engine, MinerU-Popo outperforms the baselines in both accuracy (with around 90% across subtasks) and efficiency.

**Comparison on RAG.** To evaluate the downstream utility of structured modeling, we compared RAG using MinerU-Popo to process document trees against raw OCR and visual retrieval baselines. Experiments are conducted on the five English subsets of ViDoRe V3. For tree-based retrieval, embeddings are computed solely on the concatenated title path and summary of each node. Lacking structure, the raw OCR baseline embeds full text, while the visual baseline retrieves relevant pages using ColEmbed-3B-v2 (de Souza P. Moreira et al., 2026). Finally, the retrieved contexts (nodes, text, or page images) are fed to Gemini-3-Flash for question answering.

Table 3 reports the accuracy, bounding box (bbox) recall, and latency (seconds/query) for RAG across subsets. Accuracy is evaluated via an LLM (scored 1 for completely, 0.5 for partially, and 0 for

Table 3: RAG Performance on ViDoRe V3.

| Subset | C.S. | | | Fin. | | | H.R. | | | Ind. | | | Phar. | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | Acc ↑ | Recall ↑ | Time ↓ | Acc ↑ | Recall ↑ | Time ↓ | Acc ↑ | Recall ↑ | Time ↓ | Acc ↑ | Recall ↑ | Time ↓ | Acc ↑ | Recall ↑ | Time ↓ |
| **MinerU-Popo** | **84.4** | 45.7 | 84 | 49.5 | 40.5 | 68 | **66.8** | 47.5 | 15 | 58.7 | 35.6 | 26 | **71.6** | 44.3 | 9 |
| **raw RAG** | 82.3 | 39.7 | 145 | 48.7 | 32.4 | 225 | 63.2 | 41.6 | 39 | **60.4** | 37.1 | 53 | 64.4 | 28.6 | 20 |
| **visual RAG** | 80.7 | 80.3 | 129 | **58.4** | 41.1 | 273 | 64.8 | 69.4 | 104 | 59.7 | 58.4 | 481 | 67.6 | 69.6 | 214 |

Table 4: Distribution of Question Types

| Subset | C.S. | | | Fin. | | | H.R. | | | Ind. | | | Phar. | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Category** | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| **Proportion(%)** | 0.5 | 74.4 | 25.1 | 2.3 | 41.1 | 56.6 | 2.8 | 61.3 | 35.8 | 1.4 | 35.0 | 63.6 | 2.2 | 76.9 | 20.9 |

incorrectly answered), while bbox recall is the percentage of the overlapping area of the retrieved and annotated evidence relative to the annotated evidence area. MinerU-Popo outperforms RAG on raw OCR in four of the five subsets (e.g., improving accuracy by 7.15% and bbox recall by 15.72% on the Phar. subset). Furthermore, MinerU-Popo significantly reduces latency, cutting time costs by roughly 70% and 50% on the Fin. and H.R. subsets, respectively, by efficiently embedding concise node titles and summaries. But relying on summaries for retrieval can miss fine-grained evidence, explaining its underperformance on the Ind. subset.

To further illustrate the distinctions among title, summary, and content, we employ an LLM to classify ViDoRe V3 questions. The first category queries by exact section titles or captions of images, tables, and charts. The second category includes queries targeting crucial arguments, methods, trends, and conclusions. The third category focuses on specific numerical values and highly detailed content. Table 4 shows that among questions directed at real-world documents, the first category constitute a small fraction (less than 3%). The second category (targeting crucial arguments, methods, trends, and conclusions) dominates, comprising >74.4% of the C.S. and Phar. subset and >35.0% of Ind. subset. By distilling crucial information, summaries enable efficient retrieval without repeatedly processing lengthy texts. The third category accounts for 20.9%-63.6% across subsets, where summaries yield marginal retrieval benefits since they capture high-level abstractions rather than fine-grained details.

Results in Table 3 align with the category distributions in Table 4. Specifically, 63.6% of Ind. queries require precise numerical values or fine-grained details that are often absent from high-level summaries, leading to lower accuracy when retrieval relies on node paths and summaries. In contrast, this mechanism remains effective for the other four subsets. These observations suggest that further gains may be achieved by selecting RAG strategies based on LLM-predicted query types.

Notably, while visual RAG achieves higher recall, MinerU-Popo yields superior overall QA accuracy. Visual RAG's high recall stems from page-level retrieval, which bypasses bbox alignment and guarantees 100% recall upon a successful page match. However, deriving answers directly from high-resolution, text-dense page images strains the model's visual comprehension, leading to lower accuracy in most subsets. An exception is the Fin. subset, where visual RAG achieves better answering accuracy. This is because over 25% of its queries rely on complex financial tables (versus 10% and simpler tables elsewhere). OCR struggles to parse such complex tables into OSTL or HTML format, and LLMs subsequently fail to reason over them. In contrast, VLMs process complex tables more effectively in their native image format but incurs substantial computational overhead.

**Finding 2:** MinerU-Popo improves retrieval and answering accuracy while reducing latency, highlighting the utility of post-processing in downstream tasks. The raw and visual RAG scored higher in some rare cases due to queries targeting very detailed content or complex tables.

**Comparison on End-to-End Answering.** To evaluate the representational capacity of different data formats, we conduct end-to-end question answering on MMDA across various evidence formats (Table 5). JSON and Markdown are derived from our constructed document tree, whereas SHT and XML follow ZenDB (Lin et al., 2024) and DocAgent (Sun et al., 2025), respectively.

**Finding 3:** JSON and Markdown formats derived from MinerU-Popo better capture document information for comprehensive question answering.

Table 5: End-to-End Answering Performance on MMDA.

| Type | Overall↑ | Text↑ | Hybrid↑ | Hierarchy↑ | Location↑ |
|---|---|---|---|---|---|
| **JSON** | 64.2 | 70.7 | 58.6 | 71.1 | **61.6** |
| **Markdown** | **65.5** | **75.5** | **63.2** | **81.5** | 41.7 |
| **SHT** | 60.2 | 71.0 | 54.6 | 62.0 | 49.6 |
| **XML** | 52.2 | 64.1 | 42.8 | 58.5 | 40.1 |

Table 6: Ablation Study of Dynamic Chunking and Synchronization.

| Metric | text-pre↑ | text-Rec↑ | title-TEDS↑ | image-pre↑ | image-rec↑ |
|---|---|---|---|---|---|
| **w/** | 87.8 | 93.8 | 90.6 | 87.0 | 79.5 |
| **w/o** | 84.0 | 86.6 | 85.8 | 83.7 | 75.2 |

Table 7: Ablation Study of Semantic Enrichment on ViDoRe V3 Retrieval.

| Subset | C.S. | | | Fin. | | | H.R. | | | Ind. | | | Phar. | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | IoU ↑ | Recall ↑ | Time ↓ | IoU ↑ | Recall ↑ | Time ↓ | IoU ↑ | Recall ↑ | Time ↓ | IoU ↑ | Recall ↑ | Time ↓ | IoU ↑ | Recall ↑ | Time ↓ |
| **MinerU-Popo** | 16.0 | 45.7 | 84 | 12.5 | 40.5 | 68 | 12.6 | 47.5 | 15 | 7.3 | 35.6 | 26 | 17.2 | 44.3 | 9 |
| **w/o Summary** | 15.6 | 45.7 | 194 | 12.6 | 48.7 | 248 | 12.7 | 52.4 | 47 | 7.7 | 39.9 | 73 | 18.2 | 47.6 | 24 |
| **w/o Subnode** | 14.0 | 45.6 | 54 | 10.7 | 44.3 | 32 | 9.9 | 43.3 | 10 | 6.1 | 37.7 | 20 | 16.5 | 45.4 | 5 |

We observe that tree-derived JSON and Markdown formats yield superior representations for document QA. JSON provides the most comprehensive representation by encoding both the semantics and coordinate location of document elements, achieving over 60% accuracy on MMDA. However, locations become redundant for purely semantic queries and can even impede model comprehension. Consequently, JSON underperforms Markdown on location-agnostic questions, as Markdown is more amenable to semantic reasoning. Conversely, this omission also causes Markdown to suffer a 20% performance drop on queries based on element locations. Finally, although SHT and XML are also tree-structured, they lack essential post-processing to resolve content truncations, hierarchies, and associations. This yields inferior overall performance, scoring only 50% and 60% on cross-modal and hierarchical questions respectively.

### 5.4 Ablation Study

**Dynamic Chunking and Synchronization.** Table 6 evaluates post-processing performance with and without dynamic chunking and synchronization. Processing long documents entirely degrades performance obviously (e.g., text truncation recall drops from 93.8% to 86.6%). This highlights the model's limited long-context reasoning capacity, motivating our chunk-based approach.

**Semantic Enrichment.** Table 7 ablates node chunking and summary generation in semantic enrichment. Here, IoU denotes the intersection over union between the retrieved and gold evidence bounding boxes. Replacing node summaries with full detailed contents yields higher overall IoU and Recall, but greatly increases the time cost. For instance, per-query latency on the Fin. subset jumps from 68s to 248s. Conversely, omitting node chunking accelerates retrieval due to a reduced node count. However, while larger unchunked nodes preserve more context, they introduce irrelevant noise that degrades the precision. Consequently, this omission decreases IoU across all subsets and lowers Recall on the C.S. and H.R. subsets.

## 6 Conclusion

In this paper, we propose MinerU-Popo, a document OCR post-processing pipeline. It employs a fine-tuned model to analyze table truncation, text truncation, title hierarchy, and image-text association based on OCR outputs. Dynamic chunking and synchronization enable it to adapt to long-document processing. Finally, structural and semantic enrichment produces a complete structure that is suitable for downstream tasks like RAG. Our experiments of post-processing on PostDocBench, RAG on ViDoRe V3 and end-to-end answering on MMDA confirm the effectiveness of post-processing and its beneficial role in enhancing downstream tasks.

## AI USE STATEMENT

We used AI tools in three ways. First, AI models are a core part of MinerU-Popo. We integrate MinerU, PaddleOCR, GLM-OCR, Dolphin, and MonkeyOCR as OCR backbones, fine-tune Qwen3-VL-4B for post-processing, use Qwen3-Embedding-4B for embedding-based retrieval, and use Gemini-3-Flash for final answer generation. Second, Gemini-3-Flash was used to generate initial annotations for PostDocBench, and locally deployed Qwen2.5-32B-Instruct and InternVL3-38B were used for element-wise quality checking. All annotations were refined through prompt iteration and validated by manual inspection, with over 95% agreement on a randomly sampled verification set. Third, AI tool was used for paper language polishing but not to generate our core innovation and claims. We have examined all work carried out with the help of AI and accept full responsibility for the paper's final content, including all text, claims, and artifacts.

## ETHICS STATEMENT

PostDocBench is constructed from Common Crawl (CC), a non-profit public web archive, following the data processing practices established by pioneering works in the research community. Our use of CC data is strictly for non-commercial academic research and adheres to the CC terms of use[1]; CC itself honors the robots.txt protocol at crawl time. We follow the same processing pipeline as prior landmark corpora built on CC, including T5 (Raffel et al., 2020), MMC4 (Zhu et al., 2023), OBELICS (Laurençon et al., 2023), and CCpdf (Turski et al., 2023), which focuses specifically on PDF documents from CC. In line with these established practices, we carefully filter out any documents that may contain personal information, and ensure that all data is processed in full compliance with the aforementioned standards before the final release.

## REPRODUCIBILITY STATEMENT

To make MinerU-Popo reproducible, we provide the source code of MinerU-Popo and the test JSON in the Supplementary Material. The PDFs in PostDocBench and reproduction instructions are also available at `https://anonymous.4open.science/r/popo_data-E0BB`. In addition, we have provided a detailed introduction to the process of dataset construction in Appendix A and the prompts used in Appendix B.

## REFERENCES

Common crawl terms of use, 2024. URL `https://commoncrawl.org/terms-of-use`.

A. Ben Ammar. Beyond "everything with a vision LLM": A production-ready architecture for document processing at scale, 2026. URL `https://doi.org/10.5281/zenodo.22706024`.

Jian Chen, Ruiyi Zhang, Yufan Zhou, Tong Yu, Franck Dernoncourt, Jiuxiang Gu, Ryan A Rossi, Changyou Chen, and Tong Sun. Sv-rag: Lora-contextualizing adaptation of mllms for long document understanding. *arXiv preprint arXiv:2411.01106*, 2024.

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. *arXiv preprint arXiv:2411.04952*, 2024.

Cheng Cui, Ting Sun, Suyin Liang, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Xueqing Wang, Changda Zhou, Hongen Liu, Manhui Lin, et al. Paddleocr-vl-1.5: Towards a multi-task 0.9 b vlm for robust in-the-wild document parsing. *arXiv preprint arXiv:2601.21957*, 2026.

Gabriel de Souza P. Moreira, Ronay Ak, Mengyao Xu, Oliver Holworthy, Benedikt Schifferer, Zhiding Yu, Yauhen Babakhin, Radek Osmulski, Jiarui Cai, Ryan Chesler, Bo Liu, and Even Oldridge. Nemotron colembed v2: Top-performing late interaction embedding models for visual document retrieval, 2026. URL `https://arxiv.org/abs/2602.03992`.

[1] `https://commoncrawl.org/terms-of-use`

Shuaiqi Duan, Yadong Xue, Weihan Wang, Zhe Su, Huan Liu, Sheng Yang, Guobing Gan, Guo Wang, Zihan Wang, Shengdong Yan, et al. Glm-ocr technical report. *arXiv preprint arXiv:2603.10910*, 2026.

Hao Feng, Wei Shi, Ke Zhang, Xiang Fei, Lei Liao, Dingkang Yang, Yongkun Du, Xuecheng Wu, Jingqun Tang, Yang Liu, et al. Dolphin-v2: Universal document parsing via scalable anchor prompting. *arXiv preprint arXiv:2602.05384*, 2026.

Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov, Thomas Wang, Siddharth Karamcheti, Alexander M. Rush, Douwe Kiela, Matthieu Cord, and Victor Sanh. Obelics: An open web-scale filtered dataset of interleaved image-text documents, 2023.

Yiming Lin, Madelon Hulsebos, Ruiying Ma, Shreya Shankar, Sepanta Zeigham, Aditya G Parameswaran, and Eugene Wu. Towards accurate and efficient document analytics with large language models. *arXiv preprint arXiv:2405.04674*, 2024.

António Loison, Quentin Macé, Antoine Edy, Victor Xing, Tom Balough, Gabriel Moreira, Bo Liu, Manuel Faysse, Céline Hudelot, and Gautier Viaud. Vidore v3: A comprehensive evaluation of retrieval augmented generation in complex real-world scenarios. *arXiv preprint arXiv:2601.08620*, 2026.

Jiefeng Ma, Jun Du, Pengfei Hu, Zhenrong Zhang, Jianshu Zhang, Huihui Zhu, and Cong Liu. Hrdoc: Dataset and baseline method toward hierarchical reconstruction of document structures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 1870–1877, 2023.

Linke Ouyang, Yuan Qu, Hongbin Zhou, Jiawei Zhu, Rui Zhang, Qunshu Lin, Bin Wang, Zhiyuan Zhao, Man Jiang, Xiaomeng Zhao, et al. Omnidocbench: Benchmarking diverse pdf document parsing with comprehensive annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24838–24848, 2025.

Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S Nassar, and Peter Staar. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 3743–3751, 2022.

Sundar Pichai, Demis Hassabis, and Koray Kavukcuoglu. A new era of intelligence with gemini 3. *Google. URL: https://blog.google/products-and-platforms/products/gemini/gemini-3/*, 2025.

Jake Poznanski, Aman Rangapur, Jon Borchardt, Jason Dunkelberger, Regan Huff, Daniel Lin, Christopher Wilhelm, Kyle Lo, and Luca Soldaini. olmocr: Unlocking trillions of tokens in pdfs with vision language models. *arXiv preprint arXiv:2502.18443*, 2025.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Jon Saad-Falcon, Joe Barrow, Alexa Siu, Ani Nenkova, Seunghyun Yoon, Ryan A Rossi, and Franck Dernoncourt. Pdftriage: Question answering over long, structured documents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 153–169, 2024.

Li Sun, Liu He, Shuyue Jia, Yangfan He, and Chenyu You. Docagent: An agentic framework for multi-modal long-context document understanding. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17712–17727, 2025.

Qwen Team. Qwen3 technical report, 2025. URL `https://arxiv.org/abs/2505.09388`.

Michał Turski, Tomasz Stanisławek, Karol Kaczmarek, Paweł Dyda, and Filip Graliński. Ccpdf: Building a high quality corpus for visually rich documents from web crawl data. In *International Conference on Document Analysis and Recognition*, pp. 348–365. Springer, 2023.

Bin Wang, Tianyao He, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Tao Chu, Yuan Qu, Zhenjiang Jin, Weijun Zeng, Ziyang Miao, Bangrui Xu, Junbo Niu, Mengzhang Cai, Jiantao Qiu, Qintong Zhang, Dongsheng Ma, Yuefeng Sun, Hejun Dong, Wenzheng Zhang, Jutao Xiao, Jiayong Shi, Pengyu Liao, Xiaomeng Zhao, Huaping Zhong, Liqun Wei, Jing Yu, Jie Yang, Wei Li, Shasha Wang, Qianqian Wu, Xuanhe Zhou, Weijia Li, Zhenxiang Li, Zhongying Tu, Jiang Wu, Lijun Wu, Chao Xu, Kai Chen, Wentao Zhang, Yu Qiao, Bowen Zhou, Dahua Lin, and Conghui He. Mineru2.5-pro: Pushing the limits of data-centric document parsing at scale, 2026. URL https://arxiv.org/abs/2604.04771.

Hangdi Xing, Changxu Cheng, Feiyu Gao, Zirui Shao, Zhi Yu, Jiajun Bu, Qi Zheng, and Cong Yao. Dochienet: A large and diverse dataset for document hierarchy parsing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1129–1142, 2024.

Bangrui Xu, Qihang Yao, Zirui Tang, Xuanhe Zhou, Yeye He, Shihan Yu, Qianqian Xu, Bin Wang, Guoliang Li, Conghui He, et al. Modora: Tree-based semi-structured document analysis system. *arXiv preprint arXiv:2602.23061*, 2026.

Jiarui Zhang, Yuliang Liu, Zijun Wu, Guosheng Pang, Zhili Ye, Yupei Zhong, Junteng Ma, Tao Wei, Haiyang Xu, Weikai Chen, Zeen Wang, Qiangjun Ji, Fanxi Zhou, Qi Zhang, Yuanrui Hu, Jiahao Liu, Zhang Li, Ziyang Zhang, Qiang Liu, and Xiang Bai. Monkeyocr v1.5 technical report: Unlocking robust document parsing for complex patterns, 2025a. URL https://arxiv.org/abs/2511.10390.

Qintong Zhang, Junyuan Zhang, Zhifei Ren, Linke Ouyang, Zichen Wen, Junbo Niu, Yuan Qu, Bin Wang, Ka-Ho Chow, Conghui He, et al. Docr-inspector: Fine-grained and automated evaluation of document parsing with vlm. *arXiv preprint arXiv:2512.10619*, 2025b.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025c.

Wanrong Zhu, Jack Hessel, Anas Awadalla, Samir Yitzhak Gadre, Jesse Dodge, Alex Fang, Youngjae Yu, Ludwig Schmidt, William Yang Wang, and Yejin Choi. Multimodal C4: An open, billion-scale corpus of images interleaved with text. *arXiv preprint arXiv:2304.06939*, 2023.

# APPENDIX

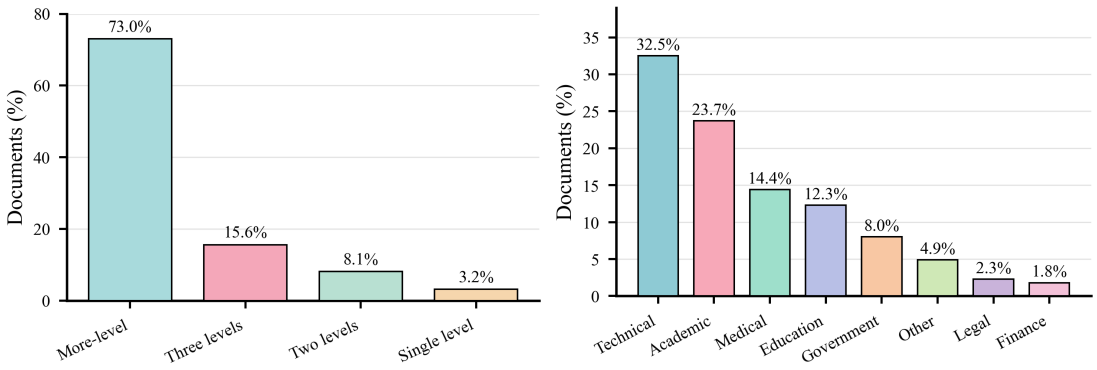
## A DATASET DETAILS

To construct PostDocBench, we first collect representative and diverse documents from large-scale real-world documents including Common Crawl (CC), a neutral, non-profit public web archive. Following the processing recognized by the academic community (Laurençon et al., 2023), we ensure that the CC data is used solely for non-commercial academic research. We strictly adhere to the terms of use (ccu, 2024) and guarantee compliance with the Robots protocol. For all collected documents, we apply a pre-filtering step to remove those containing private or sensitive information. Then, document level filtering is applied to extract the constituent documents of the dataset. As shown in Fig 4, the document-level filtering strategy provides a diverse data source and helps improve the generalization ability of the trained post-processing model. Technical reports and academic papers together account for over 50%, while medical, educational, and governmental documents also account for notable proportions. Financial reports are the least accessible among such public documents due to privacy and copyright concerns. Figure 4 also shows the annotated title hierarchy of these documents. Most documents contain headings with more than three levels. Thus the two-level (doc_title, para_title) or three_level label system used by some OCR models is inadequate.

After that, we apply task-specific filtering to OCR outputs and use Gemini-3-Flash with carefully designed prompts and structured output schemas for initial annotation (see Appendix B). During the early annotation stage, we randomly sampled 100 instances from each subtask for manual inspection. We identified several recurring annotation errors, including incorrect truncation judgments caused by ambiguous boundaries, incorrect cross-element associations, and inconsistent hierarchy

assignments. Based on these observations, we refined the annotation prompts and judgment criteria. After refinement, we further manually verified another 100 randomly sampled instances, achieving over 95% agreement. In addition, we performed element-wise verification on the annotated samples using locally deployed Qwen2.5-32B-Instruct and InternVL3-38B models for efficient large-scale quality checking. Specifically, the two models independently examine individual annotation elements, and samples that are identified as incorrect by both models are removed from the final dataset.

Finally, the dataset is randomly split into 30K training instances and 165 test instances for each of the four post-processing subtasks. Specifically, the 165 test instances of PostDocBench encompass 23,178 title elements (title hierarchy), 27,248 pairs (image-text association), 5,761 relation pairs (text truncation), and 1,317 cell-level merge units (table truncation).




(a) Title hierarchy distribution

(b) Document domain distribution

Figure 4: Dataset distribution statistics.

## B PROMPTS

We leverage LLMs for annotation, question answering and evaluation with different prompts. Here title hierarchy prompt, text truncation prompt, association prompt and table truncation prompt instruct the LLM to annotate dataset. The input is derived from task-specific filtering of MinerU predictions. QA prompt and evaluation prompt is used in our experiments of RAG and end-to-end QA.

### B.1 TITLE HIERARCHY PROMPT

```
### Instruction
Given a list of titles (marked by blue boxes in page images) in a document, please
     analyze title levels according to the page layout, visual clues and semantic
     infomation.
Identify different levels with different numbers. Use 1 for the highest level title, 2
     for the second level, and so on.
Some non-title content may be mixed into the input blocks, and you can mark them with the
     -1 level.

### Few-shot Examples of Title Hierarchy
["2000", "2010", "2020"] ---> ["#2000", "#2010", "#2020"]
["Report", "1.Challenge" , "2.Method", "3.Result"] ---> ["#Report", "##1.Challenge" ,
     "##2.Method", "##3.Result"]
["Schedule", "Day1", "Afternoon", "Night", "Day2", "Morning"] ---> ["#Schedule",
     "##Day1", "###Afternoon", "###Night", "##Day2", "###Morning"]

### Input Schema
The input is a **single JSON object** containing:

* `image`: the image of the document (served as visual evidence, and multiple pages are
     separated by black lines)
  * `blocks`: a list of titles blocks, where each element contains:
  * `idx`: unique identifier of the block (**must be referenced in the output**)
  * `content`: the textual content of the title
  * `page`: number of the page where the block is located
  * `bbox`: location of the block on the page
```

```
### Input Blocks
{{ INPUT_BLOCKS }}

### Output Schema
```json
[
  {
    idx: <int>, # The idx of the title block
    level: <int> # The level of the title (-1 if not a title)
  },
  ...
]
```

### Judgment Criteria
(1) A title, semantically, serves as a summary and overarching structure for the main
     content of a document. Visually, it should stand on its own line and may have a
     distinct font style.
(2) The title numbering in normative documents naturally indicates their hierarchy like
     1. | 1.1 | 1.2 | 2.;
(3) In page layout, the overall titles of each main block are usually parallel, while
     multiple titles within a block may be parallel or nested;
(4) Visually, multiple consecutive parallel titles usually have the same font and size,
     and larger font sizes generally indicate higher levels, but visual judgment rules
     are not always reliable;
(5) For title levels cannot be judged by title numbering, layout, and visual features,
     ultimately determine the hierarchy based on your understanding of the title and
     context semantics.

### Note
Respond strictly in the specified JSON output format. Each input block should be referred
     once in the output to identify its level.
```

### B.2 TEXT TRUNCATION PROMPT

```
### Instruction
Please identify which text blocks in the document are truncated due to column breaks,
     page breaks, inserted images or tables, and therefore need to be reconnected into a
     coherent and complete sentence.
Consider visual factors that may cause truncation, as well as grammatical correctness and
     semantic coherence from a content perspective. Output the idx of text blocks that
     you believe require reconnection and briefly explain the reason.

### Input Schema
The input is a **single JSON object** containing:

* `image`: the image of the document (served as visual evidence, and multiple pages are
     separated by black lines)
* `blocks`: a list of text blocks with potential truncation, where each element contains:
  * `idx`: unique identifier of the block (**must be referenced in the output**)
  * `content`: completed or broken sentences (the middle parts of long paragraphs are
       omitted)
  * `page`: number of the page where the block is located
  * `bbox`: location of the block on the page

### Input Blocks
{{ INPUT_BLOCKS }}

### Output Schema
```json
[
  {
    src: <int>, # The idx of the text block that needs connection at its end
    tgt: <int>, # The idx of the text block that needs connection at its start
    reason: <str> # Brief explanation. For example, 'The text in block x and block y are
         broken due to ...; the complete sentence at the connection point is ...'
  },
  ...
]
```

### Judgment Criteria
(1) The src and tgt must be two adjacent blocks in Input Blocks.
```

```
(2) Merge when two text pieces are separated due to layout factors, and the connection
     can be grammatically and semantically coherent to form a complete sentence.
(3) Truncated text blocks usually have similar font style and size.
(4) If there are a line break after standalone phrases or complete sentences between src
     and tgt, no reconnection is needed.

### Note
Respond strictly in the specified JSON output format.
```

### B.3 ASSOCIATION PROMPT

```
### Instruction
Given a list of element blocks in a document, please analyze the correlation between
     images, tables and text according to the criteria.

### Input Schema
The input is a **single JSON object** containing:

* `image`: the image of the document (served as visual evidence, and multiple pages are
     separated by black lines)
* `blocks`: a list of titles blocks, where each element contains:
* `idx`: unique identifier of the block (**must be referenced in the output**)
* `type`: the type of document elements (related to the criterias)
* `content`: the textual content of the block (None for images and tables)
* `page`: number of the page where the block is located
* `bbox`: location of the block on the page

### Input Blocks
{{ INPUT_BLOCKS }}

### Output Schema
```json
[
  {
    src: <int>, # The idx of the current block
    tgt: <int>, # The idx of the target block to which the current block should be linked
    reason: <str> # Carefully examine the reasonableness of the correlation to ensure it
         aligns with the actual document structure and the criterias
  },
  ...
]
```

### Judgment Criteria
(1) The block with type `image` or `table` should be linked to the most related `title`
     block;
(2) The block with type `image_caption` or `image_footnote` should be linked to the most
     related `image` block;
(3) The block with type `table_caption` or `table_footnote` should be linked to the most
     related `table` block;
(4) Links not falling into the above three situations cannot be connected.

### Note
Respond strictly in the specified JSON output format. You can only output links that meet
     the criterias.
```

### B.4 TABLE TRUNCATION PROMPT

```
### Instruction
Given two table fragments split across consecutive PDF pages, determine whether they
     belong to the same cross-page table and identify which columns require semantic
     merging.

### Input Schema
The input contains:

* `upper_caption`: the caption of the table on the previous page
* `upper_row`: the last few rows of the table on the previous page
* `lower_caption`: the caption of the table on the next page
* `lower_row`: the first few data rows of the table on the next page

### Input Blocks
```

```
{{ INPUT_BLOCKS }}

### Output Schema
[
   {
      judgement:<list> # A list indicating cell-level merging decision
   }
]

### Judgment Criteria
(1) If the two table fragments have different column counts or clearly cannot belong to
     the same table, output [].
(2) Otherwise, output exactly one object with a judgment list.
(3) The length of judgment must be equal to the number of columns in the table. Each
     value corresponds to one column from left to right.
(4) Use 1 if the two cells in the same column should be semantically merged into one
     logical cell, such as in cases of hyphenation, an incomplete phrase, a split
     date/number, or rowspan continuation.
(5) Use 0 if the two cells in the same column are independent row contents, even if they
     are similar or identical.
(6) For empty cells, make the decision based on the overall row context and table
     captions.

### NOTE
Respond strictly in the specified JSON output format. Do not explain your reasoning.
```

### B.5 QA PROMPT

```
You are an expert at answering queries based on documents.
Here is a list of relevant document contents: {data}
Based on the above contents, answer the following query: {query}
Keep the response short when appropriate. Output the answer only.
```

### B.6 EVALUATION PROMPT

```
You are an expert judge evaluating the accuracy of a test answer against a gold-standard
     true answer.
Your goal is to determine if the test answer captures the essential "core information."
### Evaluation Criteria:
- Correct: The test answer contains all core information of the true answer. Minor
     omissions of non-essential details or the addition of minor, non-contradictory
     information should still be marked as "Correct."
- Partially Correct: The test answer captures some of the core information, but suffers
     from significant omissions or includes substantial extra information that was not
     requested or present in the true answer.
- Incorrect: The test answer is fundamentally wrong, contradicts the true answer, or
     misses the core information entirely.
### Input Data:
Query: {query}
True Answer: {true_answer}
Test Answer: {test_answer}
### Output Format:
Provide a very brief explanation for your judgment. You must output your final response
     in a JSON format with two fields: "explanation" and "judgment" (which must be
     "Correct", "Partially Correct", or "Incorrect").
```

## C CASES

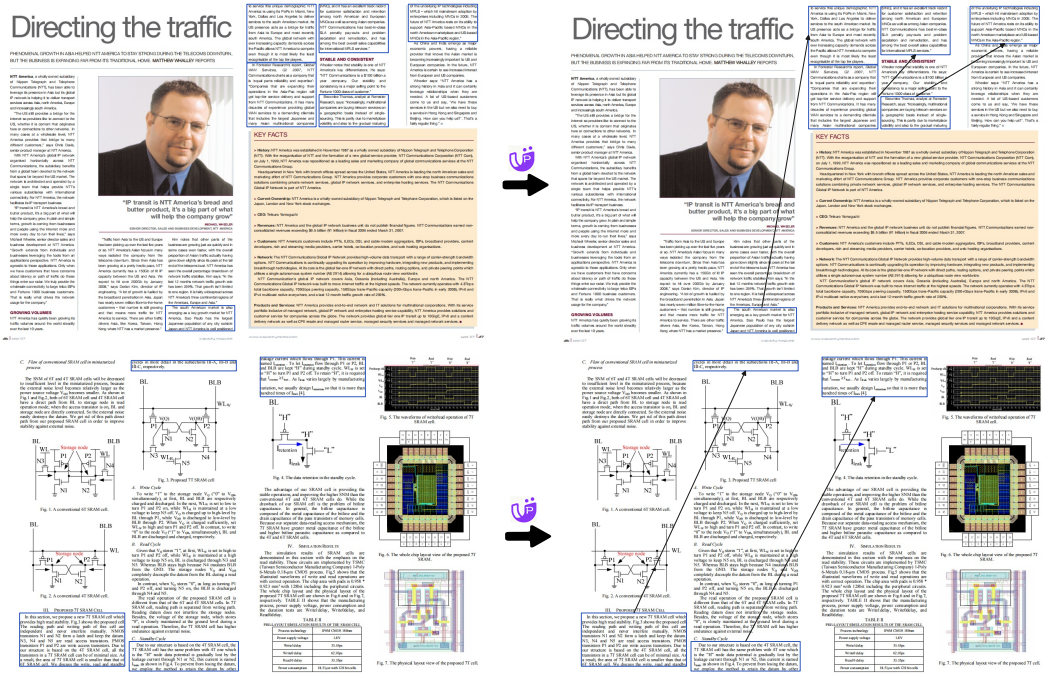



Figure 5: Cases of Text Truncation Analysis.

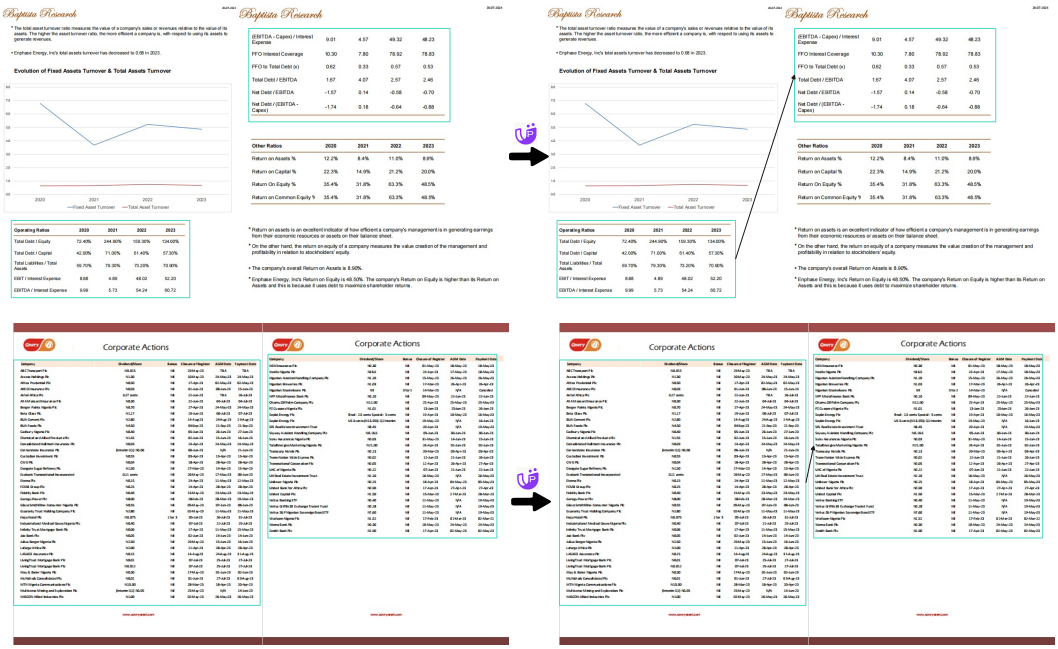



Figure 6: Cases of Table Truncation Analysis.

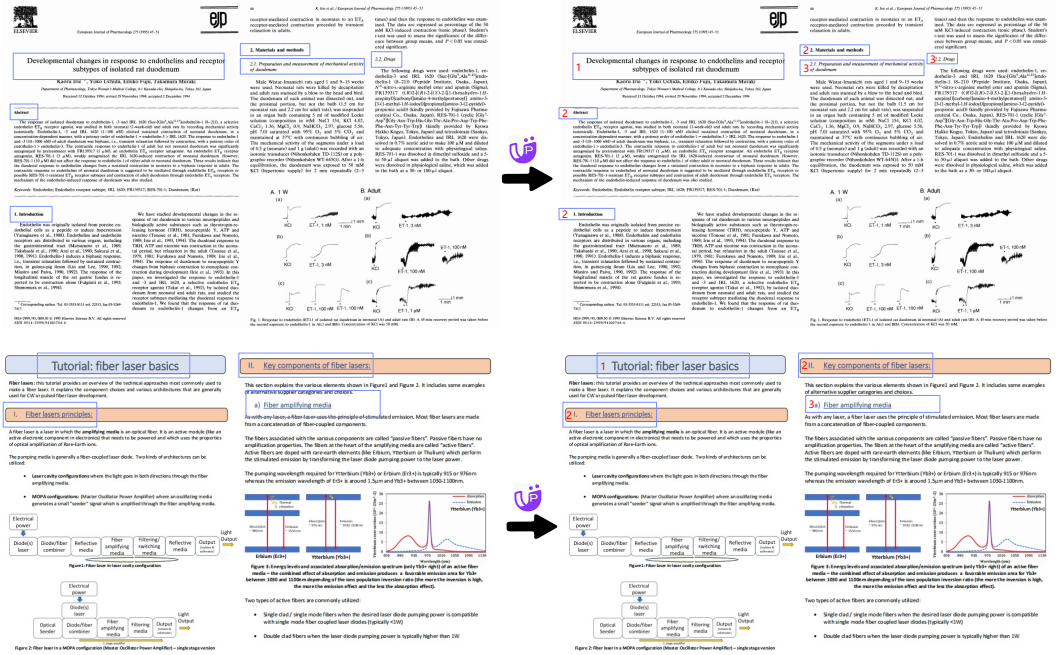


Figure 7: Cases of Title Hierarchy Analysis.

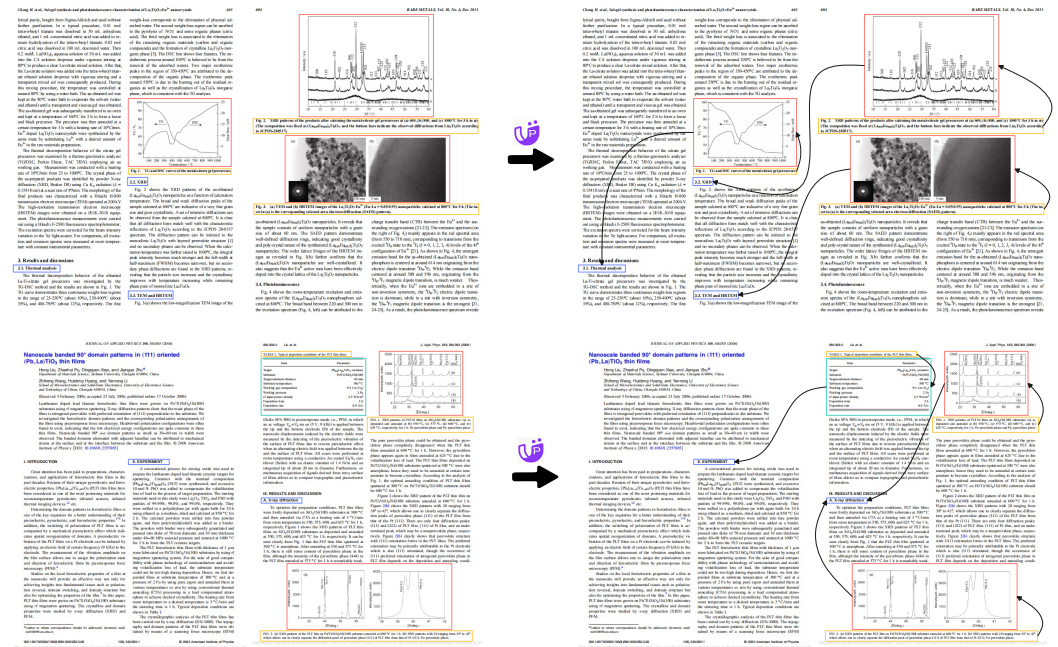


Figure 8: Cases of Text-Image Association Analysis.