MinerU: An Open-Source Solution for Precise Document Content Extraction

Bin Wang[*], **Chao Xu**[*], **Xiaomeng Zhao**, **Linke Ouyang**, **Fan Wu**, **Zhiyuan Zhao**, **Rui Xu**, **Kaiwen Liu**, **Yuan Qu**, **Fukai Shang**, **Bo Zhang**, **Liqun Wei**, **Zhihao Sui**, **Wei Li**, **Botian Shi**, **Yu Qiao**, **Dahua Lin**, **Conghui He**[†]

Shanghai Artificial Intelligence Laboratory

Abstract

Document content analysis has been a crucial research area in computer vision. Despite significant advancements in methods such as OCR, layout detection, and formula recognition, existing open-source solutions struggle to consistently deliver high-quality content extraction due to the diversity in document types and content. To address these challenges, we present MinerU, an open-source solution for high-precision document content extraction. MinerU leverages the sophisticated PDF-Extract-Kit models to extract content from diverse documents effectively and employs finely-tuned preprocessing and postprocessing rules to ensure the accuracy of the final results. Experimental results demonstrate that MinerU consistently achieves high performance across various document types, significantly enhancing the quality and consistency of content extraction. The MinerU open-source project is available at `https://github.com/opendatalab/MinerU`.

1 Introduction

The release of ChatGPT [23; 24] at the end of 2022 ignites a wave of interest in the research and application of large language models (LLMs) [15; 27; 29; 41; 6; 30; 7; 5; 1; 12]. Central to training high-quality LLMs is the acquisition and construction of high-quality data. As LLMs rapidly evolve, data from internet web pages is becoming insufficient to support further improvements in model training. Document data, which contains a wealth of knowledge, emerges as a crucial resource for enhancing LLMs. The introduction and development of Retrieval-Augmented Generation (RAG) [14; 26; 2; 8] in 2023 further intensify the demand for high-quality document extraction in both industry and academia.

Currently, there are four main technical approaches to document content extraction:

OCR-based Text Extraction. This approach uses OCR models to directly extract text from documents. While feasible for plain text documents, it introduces significant noise when documents contain images, tables, formulas, and other elements, rendering it unsuitable for high-quality data extraction.

Library-based Text Parsing. For non-scanned documents, open-source Python libraries such as PyMuPDF can directly read content without invoking OCR, offering faster and more accurate text results. However, this approach fails when documents contain formulas, tables, and other elements.

Multi-Module Document Parsing. This approach employs various document parsing models to process document images in multiple stages. Initially, layout detection algorithms identify different

[*]Project leader.
[†]Corresponding author: heconghui@pjlab.org.cn

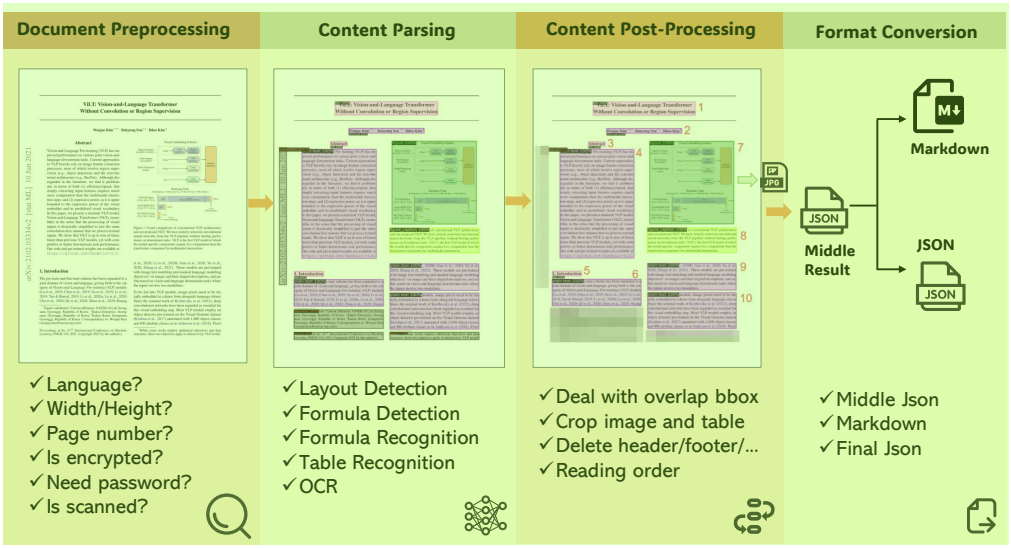

Figure 1: Overview of the MinerU framework processing workflow.

types of regions, such as images, image captions, tables, table captions, headings, and text. Subsequently, different recognizers are applied to these specific regions. For instance, OCR is used for text and headings, formula recognition models handle formulas, and table recognition models process tables. Although this method is theoretically capable of producing high-quality document results, existing open-source models often focus solely on academic papers and perform poorly on diverse document types, including textbooks, exam papers, research reports, and newspapers.

End-to-End MLLM Document Parsing. With the advancement of multimodal large language models (MLLMs), numerous models for document content extraction emerge, such as Donut [13], Nougat [4], Kosmos-2.5 [21], Vary [34], Vary-toy [35], mPLUG-DocOwl-1.5 [9], mPLUG-DocOwl2 [10], Fox [17], and GOT [36]. These models benefit from continuously optimized encoders (e.g., SwinTransformer [20], ViTDet [16]) and decoders (e.g., mBART [18], Qwen2-0.5B [38]) as well as data engineering, gradually improving extraction performance. However, they still face challenges related to data diversity and high inference costs.

To better extract diverse documents while ensuring low inference costs and high inference quality, we propose MinerU, an all-in-one document extraction tool. MinerU's primary technical approach is based on the multi-module document parsing strategy. Unlike existing document parsing algorithms, MinerU leverages various open-source models from the PDF-Extract-Kit[3], which are trained on diverse real-world documents to achieve high-quality results in tasks involving complex layouts and intricate formulas. After obtaining the positions and recognition content of different regions from the models, MinerU employs a tailored processing workflow to ensure the accuracy of the results.

Using MinerU for document extraction offers several advantages:

- **Adaptability to Diverse Document Layouts**: Supports a wide range of document types, including but not limited to academic papers, textbooks, exam papers, and research reports.
- **Content Filtering**: Allows filtering of irrelevant regions such as headers, footers, footnotes, and side notes, enhancing document readability.
- **Accurate Segmentation**: Combines model-based and rule-based post-processing for paragraph recognition, enabling cross-column and cross-page paragraph merging.
- **Robust Page Element Recognition**: Accurately distinguishes between formulas, tables, images, text blocks, and their respective captions.

[3] https://github.com/opendatalab/PDF-Extract-Kit

2 MinerU Framework

As shown in Figure 1, MinerU processes diverse user-input PDF documents and converts them into desired machine-readable formats (Markdown or JSON) through a series of steps. Specifically, the processing workflow of MinerU is divided into four stages:

Document Preprocessing. This stage uses PyMuPDF[4] to read PDF files, filter out unprocessable files (e.g., encrypted files), and extract PDF metadata, including the document's parseability (categorized into parseable and scanned PDFs), language type, and page dimensions.

Document Content Parsing. This stage employs the PDF-Extract-Kit, a high-quality PDF document extraction algorithm library, to parse key document contents. It begins with layout analysis, including layout and formula detection. Different recognizers are then applied to various regions: OCR [28; 19] for text and titles, formula recognition [3; 25; 33] for formulas, and table recognition [37] for tables.

Document Content Post-Processing. Based on the outputs from the second stage, this stage removes invalid regions, stitches content according to regional positioning information, and ultimately obtains the positioning, content, and sorting information for different document regions.

Format Conversion. Based on the results of document post-processing, various formats required by users, such as Markdown, can be generated for subsequent use.

2.1 Document Preprocessing

PDF document preprocessing has two main objectives: first, to filter out unprocessable PDFs, such as non-PDF files, encrypted documents, and password-protected documents. Second, to obtain PDF metadata for subsequent use. The acquisition of PDF metadata includes the following aspects:

- **Language Identification**: Currently, MinerU identifies and processes only Chinese and English documents. The language type needs to be specified as a parameter when performing OCR, and the quality of processing for other languages is not guaranteed.
- **Content Garbled Detection**: Some text-based PDFs contain text that appears garbled when copied. Such PDFs need to be identified in advance so that OCR can be used for text recognition in the next step.
- **Scanned PDF Identification**: For text-based PDFs (as opposed to scanned PDFs), MinerU directly uses PyMuPDF for text extraction. However, for scanned PDFs, OCR needs to be enabled. Scanned PDFs are identified based on characteristics such as a larger image area compared to the text area, sometimes covering the entire PDF page, and an average text length per page close to zero.
- **Page Metadata Extraction**: Extracts document metadata such as total page count, page dimensions (width and height), and other relevant attributes.

2.2 Document Content Parsing

In the document parsing stage, MinerU uses the PDF-Extract-Kit model library to detect different types of regions and recognize the corresponding region contents (OCR, formula recognition, table recognition, etc.). PDF-Extract-Kit is an algorithm library for PDF parsing, containing various state-of-the-art (SOTA) open-source PDF document parsing algorithms. Unlike other open-source algorithm libraries, PDF-Extract-Kit aims to build a model library that ensures accuracy and speed when dealing with diverse data in real-world scenarios. When the SOTA open-source algorithms in a specific field fail to meet practical needs, PDF-Extract-Kit employs data engineering to construct high-quality, diverse datasets for further model fine-tuning, thereby significantly enhancing the model's robustness to varied data. The current version of MinerU[5] utilizes five models: layout detection, formula detection, table recognition, formula recognition and OCR.

[4] https://github.com/pymupdf/PyMuPDF
[5] Current version: v0.8.1

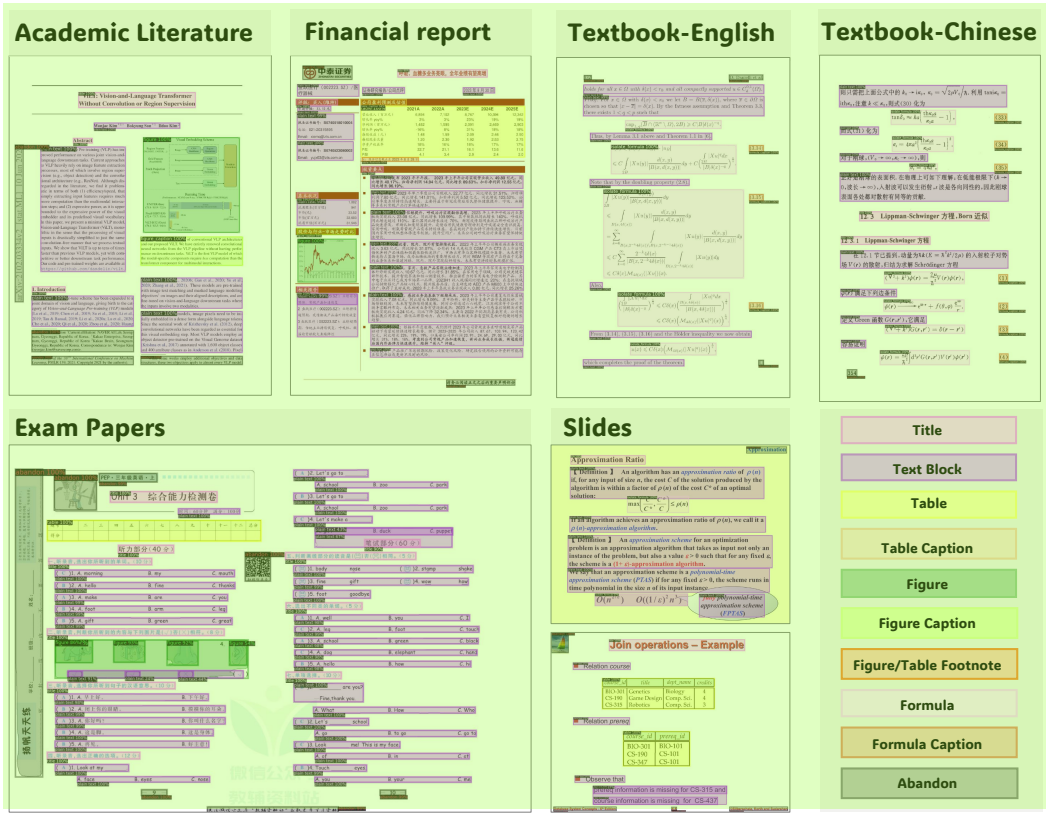

Figure 2: High-quality layout detection results on diverse documents.

2.2.1 Layout Analysis

Layout analysis is the crucial first step in document parsing, aiming to distinguish different types of elements and their corresponding regions on a page. Existing layout detection algorithms [11; 39] perform well on paper-type documents but struggle with diverse documents such as textbooks and exam papers. Therefore, PDF-Extract-Kit constructs a diverse layout detection training set and trains high-quality models for document region extraction.

The data engineering-based model training approach is as follows:

- **Diverse Data Selection**: Collects diverse PDF documents, clusters them based on visual features, and samples data from different cluster centers to obtain an initial diverse document dataset. The categories include scientific papers, general books, textbooks, exam papers, magazines, PPTs, research reports, etc.
- **Data Annotation**: Categorizes the layout annotation types involved in the document components, including titles, body paragraphs, images, image captions, tables, table captions, image table notes, inline formulas, formula labels, and discard types (such as headers, footers, page numbers, and page notes). Detailed annotation standards are established for each type, and approximately 21K data points are annotated as the training set.
- **Model Training**: Fine-tunes the model for the Layout Detection task based on the layout detection models [11; 31]. The number of classes parameter is modified to align with our categorized layout types.
- **Iterative Data Selection and Model Training**: During model iteration, partitions a portion of the data as a validation set and uses its results to guide the focus of subsequent data iterations. If a specific category from a particular source of PDF documents scores low, the sampling weight for PDF pages containing that specific category from that source is increased in the next iteration, thereby more efficiently iterating the data and model.

The model trained on diverse datasets performs significantly better on varied documents. As shown in Figure 2, the layout detection model trained on diverse layout detection data performs well on documents such as textbooks, far exceeding the performance of open-source SOTA models.

2.2.2 Formula Detection

Layout analysis can accurately locate most elements in a document, but formula types, especially inline formulas, can be visually indistinguishable from text, such as "$100cm^2$" and "$(\alpha_1, \alpha_2, \ldots, \alpha_n)$". If formulas are not detected in advance, subsequent text extraction using OCR or Python libraries may result in garbled text, affecting the overall accuracy of the document, which is crucial for scientific documents. Therefore, we trained a dedicated formula detection model.

For the formula detection dataset annotation, we defined three categories: inline formulas, displayed formulas, and an ignore class. The ignore class mainly refers to areas that are difficult to determine as formulas, such as "50%", "NaCl", and "1-2 days". Ultimately, we annotated 24,157 inline formulas and 1,829 displayed formulas on 2,890 pages from Chinese and English papers, textbooks, books, and financial reports for training.

After obtaining a diverse formula detection dataset, PDF-Extract-Kit trains a YOLO-based model, which performs well in terms of speed and accuracy on various documents.

2.2.3 Formula Recognition

Varied documents contain various types of formulas, such as short printed inline formulas and complex displayed formulas. Some documents are scanned, leading to noisy formula content and even the presence of handwritten formulas. Therefore, MinerU employs the self-developed UniMERNet [32] model for formula recognition. The UniMERNet model is trained on the large-scale diverse formula recognition dataset UniMER-1M. Thanks to the optimization of the model structure, it achieves good performance on various types of formulas (SPE, CPE, SCE, HWE) in real-world scenarios, comparable to commercial software MathPix [22; 33].

2.2.4 Table Recognition

Tables serve as an effective way to present structured data across various contexts, including scientific publications, financial reports, invoices, web pages, and beyond. Extracting tabular data from visual table images, known as the table recognition task, is challenging primarily because tables often contain complex column and row headers, as well as spanning cell operations. By leveraging MinerU, users can perform Table-to-LaTex or Table-to-HTML tasks to extract structured data from tables. MinerU employs TableMaster [40] and StructEqTable[6] for performing the table recognition task. TableMaster is trained using PubTabNet dataset (v2.0.0) [42] while StructEqTable is trained using data from DocGenome benchmark [37]. TableMaster divides the table recognition task into four sub-tasks including table structure recognition, text line detection, text line recognition, and box assignment, while StructEqTable performs the table recognition task in an end-to-end manner, demonstrating stronger recognition performance and delivering good results even with complex tables.

2.2.5 OCR

After excluding special regions (tables, formulas, images, etc.) in the document, we can directly apply OCR to recognize text regions. MinerU uses Paddle-OCR[7] integrated into PDF-Extract-Kit for text recognition. However, as shown in Figure 3, direct OCR on the entire page can sometimes result in text from different columns being recognized as a single column, which leads to incorrect text order. Therefore, we perform OCR based on the text regions (titles, text paragraphs) detected by the layout analysis to avoid disrupting the reading order.

As shown in Figure 4, When performing OCR on text blocks with inline formulas, we first mask the formulas using the coordinates provided by the formula detection model, then perform OCR, and finally reinsert the formulas back into the OCR results.

[6] https://github.com/UniModal4Reasoning/StructEqTable-Deploy
[7] https://github.com/PaddlePaddle/PaddleOCR

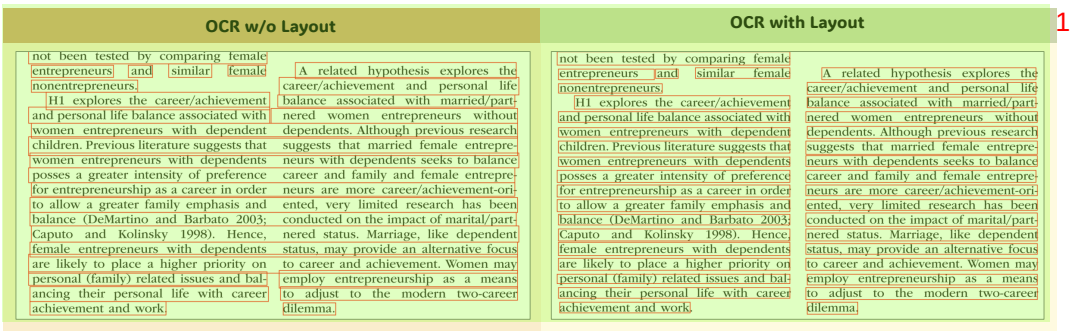

Figure 3: OCR results comparison on a multi-column document. The left image shows incorrect text order without layout detection, while the right image preserves the correct order with layout detection.

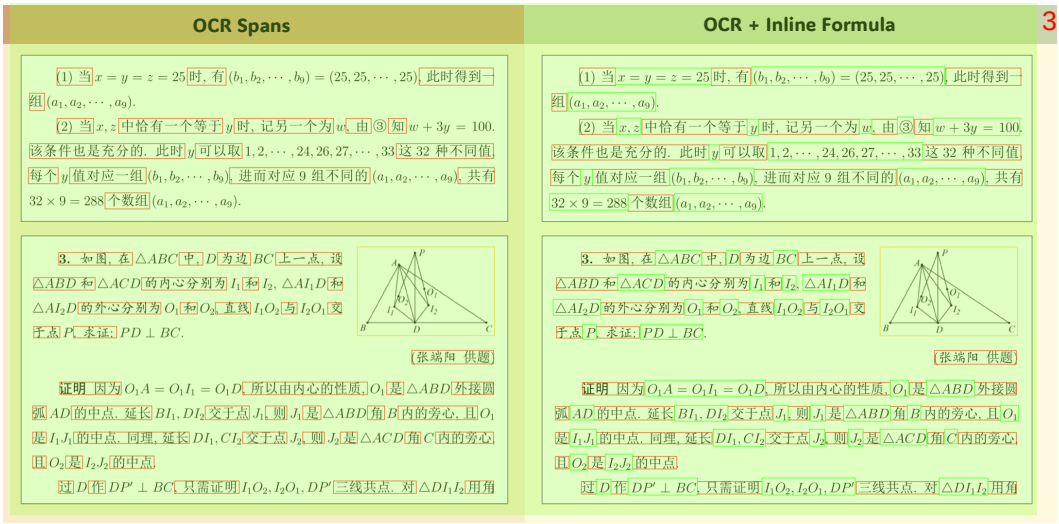

Figure 4: OCR results for text blocks with inline formulas. The left image shows OCR results with formulas masked, and the right image shows the final results with formulas reintegrated.

2.3 Document Content Post-Processing

The post-processing stage primarily addresses the issue of content ordering. Due to potential overlaps among text, images, tables, and formula boxes output by the model, as well as frequent overlaps among text lines obtained through OCR or API, sorting the text and elements poses a significant challenge. This stage focuses on handling the relationships between Bounding Boxes (BBox). Figure 5 shows a visualization of the results before and after resolving overlapping bounding boxes.

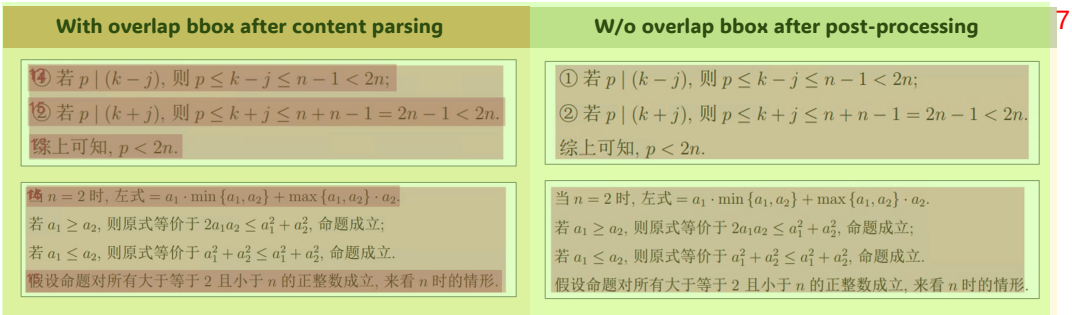

Figure 5: Bounding Boxes before and after resolving overlaps. The left image shows overlapping BBoxes, and the right image shows the results after removing overlaps.

Figure 6: Visualization of the region sorting results.

The solutions to the BBox relationships include the following aspects:

Containment Relationships. Remove formulas and text blocks contained within image and table regions, as well as boxes contained within formula boxes.

Partial Overlap Relationships. Partially overlapping text boxes are shrunk vertically and horizontally to avoid mutual coverage, ensuring that the final position and content remain unaffected, which facilitates subsequent sorting operations. For partial overlaps between text and tables/images, the integrity of the text is ensured by temporarily ignoring the images and tables.

After addressing the nested and partially overlapping BBoxes, MinerU developed a segmentation algorithm based on the human reading order, "top to bottom, left to right." This algorithm divides the entire page into several regions, each containing multiple BBoxes, while ensuring that each region contains at most one column. This ensures that the text is read line by line from top to bottom, adhering to the natural human reading sequence. The segmented groups are then sorted according to their positional relationships, determining the reading order of each element within the PDF.

2.4 Format Conversion

To accommodate varying user requirements for output formats, MinerU stores the processed PDF data in an intermediate structure. The intermediate structure is a large JSON file,with the most important fields listed in Table 1.

Field Name	Function
pdf_info	This field contains multiple subfields. The most important one is para_blocks, an ordered array where each element represents a segment of content on the PDF, which can be images, image captions, text, titles, tables, etc. Concatenating the content of this array in order reconstructs the content of the PDF (excluding headers, footers, page numbers, etc.).
_parse_type	Takes values of txt or ocr. If it is txt, it means the text is directly extracted from the PDF via API. If it is ocr, it means the text is obtained through an OCR engine.
_version_name	The software version, which can be used to track errors in data processing.

Table 1: Important Fields in the Intermediate Structure

MinerU's command line supports output in Markdown and a custom JSON format, both converted from the aforementioned intermediate structure. During the format conversion process, images, tables, and other elements can be cropped as needed. For detailed format descriptions, refer to the documentation[8].

Category	Description
Research Report	Financial reports from the internet, featuring large tables, complex merged tables, horizontal tables mixed with text, single and double columns, and complex layouts.
Standard Textbook	Textbooks from the internet, characterized by single-column layout, black-and-white color, nested complex formulas, and large matrices.
Special Image-Text Textbook	Textbooks from the internet with special image-text content, covering subjects like English, Mathematics, and Chinese (including Pinyin).
Academic Paper	Documents from arXiv and SCIHUB, featuring complex layouts with single and double columns, figures, tables, and formulas.
Picture Album	Picture albums from the internet, characterized by pages with large images.
PowerPoint Slides	PDF files converted from internet PowerPoint slides, featuring background colors and covering subjects like Biology, Chinese, English, and Physics.
Standard Exam Paper	Exam papers from the internet, characterized by exam layout, black-and-white background, and covering subjects like Computer Science, Mathematics, and Chinese, including primary, middle, high school, and industry question banks.
Special Image-Text Exam Paper	Exam papers from the internet with special image-text content, covering subjects like English, Mathematics, and Chinese (including Pinyin).
Historical Document	Documents from the internet, characterized by vertical layout, right-to-left reading order, and traditional Chinese fonts.
Notes	Notes from the internet, featuring handwritten content, including notes from three middle school students.
Standard Book	Books from the internet, characterized by single-column layout and black-and-white background.

Table 2: Categories of Documents and Their Descriptions

3 MinerU Quality Assessment

To assess the quality of content extracted by MinerU from PDFs, we explore two dimensions. First, we conduct a standalone evaluation of the core modules responsible for document content parsing to ensure the accuracy of model inference results. The quality of model results is crucial for the final content quality, as evidenced by the overall process. At this stage, we specifically evaluate three modules: layout detection, formula detection, and formula recognition. We construct a diverse evaluation dataset and compare the performance of the core algorithm components of MinerU's

[8] https://github.com/opendatalab/MinerU/blob/master/docs/output_file_en_us.md

PDF-Extract-Kit with other state-of-the-art (SOTA) open-source models. Additionally, we perform manual quality checks to assess MinerU's performance on diverse document types.

3.1 Construction of a Diverse Evaluation Dataset

To assess the quality of document content extraction in real-world scenarios, we initially constructed a diverse evaluation dataset for model assessment and visual analysis of extracted content. As shown in Table 2, the diverse dataset includes 11 types of documents, from which we further construct evaluation datasets for layout detection and formula detection.

Model	Academic Papers Val			Textbook Val		
	mAP	AP50	AR50	mAP	AP50	AR50
DocXchain	52.8	69.5	77.3	34.9	50.1	63.5
Surya	24.2	39.4	66.1	13.9	23.3	49.9
360LayoutAnalysis-Paper	37.7	53.6	59.8	20.7	31.3	43.6
360LayoutAnalysis-Report	35.1	46.9	55.9	25.4	33.7	45.1
LayoutLMv3-Finetined (Ours)	**77.6**	**93.3**	**95.5**	**67.9**	**82.7**	**87.9**

Table 3: Performance of different models on layout detection

3.2 Evaluation of Core Algorithm Modules

3.2.1 Layout Detection

We compared MinerU's layout detection model with existing open-source models, including DocXchain [39], Surya[9], and two models from 360LayoutAnalysis[10]. Table 3 shows the performance of each model on academic papers and textbook validation sets. The LayoutLMv3-SFT model, as shown in the table, was fine-tuned on our internally constructed layout detection dataset based on the LayoutLMv3-base-chinese pretrained model. The initial evaluation dataset for layout detection includes validation sets from academic papers and textbooks.

Model	Academic Papers Val		Multi-source Val	
	AP50	AR50	AP50	AR50
Pix2Text-MFD	60.1	64.6	58.9	62.8
YOLOv8-Finetined (Ours)	**87.7**	**89.9**	**82.4**	**87.3**

Table 4: Performance of different models on formula detection

3.2.2 Formula Detection

We compare MinerU's formula detection model with the open-source formula detection model Pix2Text-MFD. Additionally, YOLO-Finetuned is a model we trained based on YOLOv8 using a diverse formula detection training set.

The formula detection evaluation dataset comprises pages from academic papers and various sources for formula detection. The results, as shown in Table 4, demonstrate that the detection model fine-tuned on diverse data significantly outperforms previous open-source models on both papers and various other document types.

3.2.3 Formula Recognition

PDFs contain various types of formulas, and to achieve robust formula recognition results on diverse formulas, we use UniMERNet as our formula recognition model. Given that the same formula may have various expressions, we utilize CDM [33] for evaluating formula recognition performance. As

[9] https://github.com/VikParuchuri/surya
[10] https://github.com/360AILAB-NLP/360LayoutAnalysis

Model	ExpRate	ExpRate@CDM	BLEU	CDM
Pix2tex	0.1237	0.291	0.4080	0.636
Texify	0.2288	0.495	0.5890	0.755
Mathpix	0.2610	0.5	0.8067	0.951
UniMERNet	0.4799	0.811	0.8425	0.968

Table 5: Evaluation results of different models on the UniMER-Test dataset. Results are adapted from the CDM paper [33]. The ExpRate and BLEU metrics are shown in gray as they are considered less reliable. The CDM metric is unaffected by the diversity of formula representations and is therefore a more reasonable metric for comparing the formula recognition performance of different models.

shown in Table 5, UniMERNet's formula recognition capability far surpasses that of other open-source models and is comparable to commercial software like Mathpix.

Based on the above evaluations, we can conclude that the models used by MinerU, trained specifically on diverse document sources, significantly outperform other open-source models designed for single document types, ensuring the accuracy of parsing results.

3.3 End-to-End Results Visualization and Analysis

To assess the quality of MinerU's final extraction results, in addition to ensuring the quality of the model extraction results mentioned above, we also perform post-processing on the extracted results, such as removing noise content and stitching model outputs. MinerU's post-processing operations ensure the readability and accuracy of the final results. As shown in Figure 7, MinerU achieves excellent extraction results on diverse documents.

From the visualization results, it is evident that the layout detection results accurately locate different regions. The spans [11] show that the formula detection and OCR detection results are satisfactory, ultimately stitching together into high-quality Markdown results.

4 Conclusion and Future Work

In this work, we introduce MinerU, a one-stop PDF document extraction tool. Thanks to high-quality model inference results and meticulous pre-processing and post-processing operations, MinerU ensures high-quality extraction results even when dealing with diverse document types. Although MinerU has demonstrated significant advantages, there is still ample room for improvement. Moving forward, we continuously upgrade MinerU in the following areas:

- **Enhancement of Core Components**. We will iteratively update the existing models in the PDF-Extract-Kit to further improve the extraction quality for diverse documents. Additionally, we will introduce new models, such as table recognition and reading order, to enhance MinerU's overall capabilities.
- **Improvement of Usability and Inference Speed**. We will further optimize MinerU's processing pipeline to accelerate document extraction speed and enhance usability. Moreover, we will deploy more efficient online inference services to meet users' real-time needs.
- **Systematic Benchmark Construction**. We will establish a systematic evaluation benchmark for diverse documents to clearly compare the results of MinerU with those of state-of-the-art open-source methods, aiding community users in selecting the most suitable models for their needs.

[11] In document extraction tasks, a span is often used to mark and process specific text segments

Figure 7: Visualization of MinerU's extraction process on various document types. From left to right: layout detection results, span results, and final Markdown results.

References

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

[2] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2023.

[3] Lukas Blecher. pix2tex - latex ocr. `https://github.com/lukas-blecher/LaTeX-OCR`, 2022. Accessed: 2024-2-29.

[4] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural optical understanding for academic documents. *arXiv preprint arXiv:2308.13418*, 2023.

[5] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[6] Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*, 2024.

[7] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

[9] Anwen Hu, Haiyang Xu, Jiabo Ye, Ming Yan, Liang Zhang, Bo Zhang, Chen Li, Ji Zhang, Qin Jin, Fei Huang, et al. mplug-docowl 1.5: Unified structure learning for ocr-free document understanding. *arXiv preprint arXiv:2403.12895*, 2024.

[10] Anwen Hu, Haiyang Xu, Liang Zhang, Jiabo Ye, Ming Yan, Ji Zhang, Qin Jin, Fei Huang, and Jingren Zhou. mplug-docowl2: High-resolution compressing for ocr-free multi-page document understanding. *arXiv preprint arXiv:2409.03420*, 2024.

[11] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4083–4091, 2022.

[12] AQ Jiang, A Sablayrolles, A Mensch, C Bamford, DS Chaplot, D de las Casas, F Bressand, G Lengyel, G Lample, L Saulnier, et al. Mistral 7b (2023). *arXiv preprint arXiv:2310.06825*, 2023.

[13] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pages 498–517. Springer, 2022.

[14] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[15] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023.

[16] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. In *European conference on computer vision*, pages 280–296. Springer, 2022.

[17] Chenglong Liu, Haoran Wei, Jinyue Chen, Lingyu Kong, Zheng Ge, Zining Zhu, Liang Zhao, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. Focus anywhere for fine-grained multi-page document understanding. *arXiv preprint arXiv:2405.14295*, 2024.

[18] Y Liu. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*, 2020.

[19] Yuliang Liu, Zhang Li, Biao Yang, Chunyuan Li, Xucheng Yin, Cheng-lin Liu, Lianwen Jin, and Xiang Bai. On the hidden mystery of ocr in large multimodal models. *arXiv preprint arXiv:2305.07895*, 2023.

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[21] Tengchao Lv, Yupan Huang, Jingye Chen, Lei Cui, Shuming Ma, Yaoyao Chang, Shaohan Huang, Wenhui Wang, Li Dong, Weiyao Luo, et al. Kosmos-2.5: A multimodal literate model. *arXiv preprint arXiv:2309.11419*, 2023.

[22] Mathpix. Mathpix. `https://mathpix.com/`. Accessed: 2024-8-15.

[23] OpenAI. Chatgpt. `https://openai.com/blog/chatgpt`, 2023.

[24] OpenAI. Gpt-4 technical report, 2023.

[25] Vik Paruchuri. Texify. `https://github.com/VikParuchuri/texify`, 2023. Accessed: 2024-2-29.

[26] Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.

[27] Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

[28] Ray Smith. An overview of the tesseract ocr engine. In *Ninth international conference on document analysis and recognition (ICDAR 2007)*, volume 2, pages 629–633. IEEE, 2007.

[29] InternLM Team. Internlm: A multilingual language model with progressively enhanced capabilities, 2023.

[30] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[31] Ao Wang, Hui Chen, Lihao Liu, Kai Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Yolov10: Real-time end-to-end object detection. *arXiv preprint arXiv:2405.14458*, 2024.

[32] Bin Wang, Zhuangcheng Gu, Chao Xu, Bo Zhang, Botian Shi, and Conghui He. Unimernet: A universal network for real-world mathematical expression recognition. *arXiv preprint arXiv:2404.15254*, 2024.

[33] Bin Wang, Fan Wu, Linke Ouyang, Zhuangcheng Gu, Rui Zhang, Renqiu Xia, Bo Zhang, and Conghui He. Cdm: A reliable metric for fair and accurate formula recognition evaluation. *arXiv preprint arXiv:2409.03643*, 2024.

[34] Haoran Wei, Lingyu Kong, Jinyue Chen, Liang Zhao, Zheng Ge, Jinrong Yang, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. Vary: Scaling up the vision vocabulary for large vision-language models. *arXiv preprint arXiv:2312.06109*, 2023.

[35] Haoran Wei, Lingyu Kong, Jinyue Chen, Liang Zhao, Zheng Ge, En Yu, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. Small language model meets with reinforced vision vocabulary. *arXiv preprint arXiv:2401.12503*, 2024.

[36] Haoran Wei, Chenglong Liu, Jinyue Chen, Jia Wang, Lingyu Kong, Yanming Xu, Zheng Ge, Liang Zhao, Jianjian Sun, Yuang Peng, et al. General ocr theory: Towards ocr-2.0 via a unified end-to-end model. *arXiv preprint arXiv:2409.01704*, 2024.

[37] Renqiu Xia, Song Mao, Xiangchao Yan, Hongbin Zhou, Bo Zhang, Haoyang Peng, Jiahao Pi, Daocheng Fu, Wenjie Wu, Hancheng Ye, et al. Docgenome: An open large-scale scientific document benchmark for training and testing multi-modal large language models. *arXiv preprint arXiv:2406.11633*, 2024.

[38] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

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

[40] Jiaquan Ye, Xianbiao Qi, Yelin He, Yihao Chen, Dengyi Gu, Peng Gao, and Rong Xiao. Pingan-vcgroup's solution for icdar 2021 competition on scientific literature parsing task b: table recognition to html. *arXiv preprint arXiv:2105.01848*, 2021.

[41] Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, et al. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*, 2024.

[42] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: data, model, and evaluation. In *European conference on computer vision*, pages 564–580. Springer, 2020.