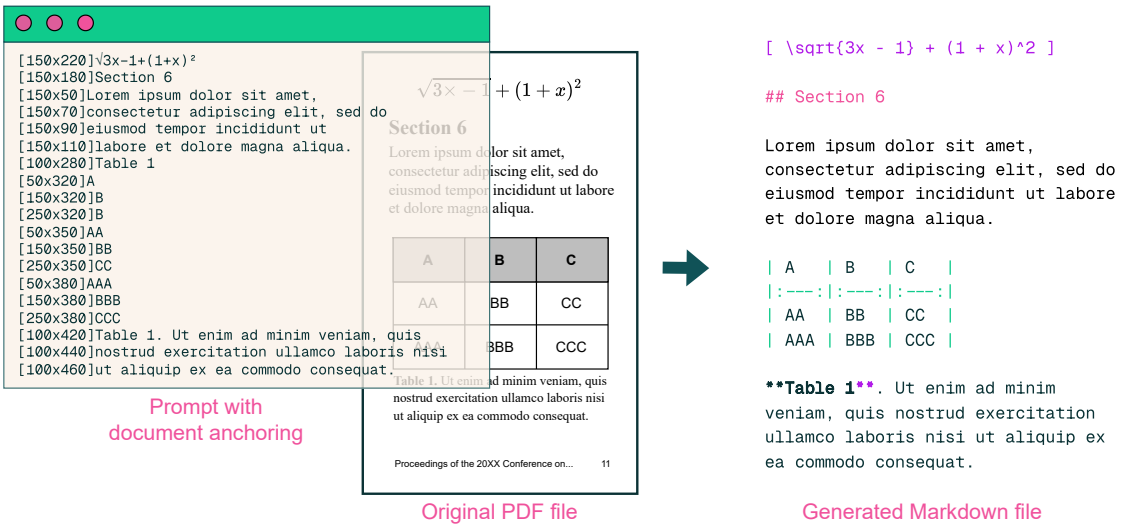
olmOCR: Unlocking Trillions of Tokens in PDFs with Vision Language Models

Jake Poznanski♥

Jason Dunkelberger **Regan Huff** **Daniel Lin** **Aman Rangapur** **Christopher Wilhelm**

Kyle Lo♥ **Luca Soldaini**♥

Allen Institute for AI, Seattle, USA {jakep|kylel|lucas}@allenai.org ♥ indicates core contributors.



Abstract

PDF documents have the potential to provide trillions of novel, high-quality tokens for training language models. However, these documents come in a diversity of types with differing formats and visual layouts that pose a challenge when attempting to extract and faithfully represent the underlying content for language model use. We present OLMOCR, an open-source Python toolkit for processing PDFs into clean, linearized plain text in natural reading order while preserving structured content like sections, tables, lists, equations, and more. Our toolkit runs a fine-tuned 7B vision language model (VLM) trained on a sample of 260,000 pages from over 100,000 crawled PDFs with diverse properties, including graphics, handwritten text and poor quality scans. OLMOCR is optimized for large-scale batch processing, able to scale flexibly to different hardware setups and convert a million PDF pages for only $190 USD. We release all components of OLMOCR including VLM weights, data and training code, as well as inference code built on serving frameworks including vLLM and SGLang.

Code	allenai/olmocr
Weights & Data	allenai/olmocr
Demo	olmocr.allenai.org

1 Introduction

Access to clean, coherent textual data is a crucial component in the life cycle of modern language models (LMs). During model development, LMs require training on trillions of tokens derived from billions of documents (Soldaini et al., 2024; Penedo et al., 2024; Li et al., 2024); errors from noisy or low fidelity content extraction and representation can result in training instabilities or even worse downstream performance (Penedo et al., 2023; Li et al., 2024; OLMo et al., 2024). During inference, LMs are often prompted with plain text representations of relevant document context to ground user prompts; for example, consider information extraction (Kim et al., 2021) or AI reading assistance (Lo et al., 2024) over a user-provided document and cascading downstream errors due to low quality representation of the source document.

While the internet remains a valuable source of textual content for language models, large amounts of content are not readily available through web pages. Electronic documents (*e.g.*, PDF, PS, DjVu formats) and word processing files (*e.g.*, DOC, ODT, RTF) are widely-used formats to store textual content. However, these formats present a unique challenge: unlike modern web standards, they encode content to facilitate rendering on fixed-size physical pages, at the expense of preserving logical text structure. For example, consider the PDF format, which originated as a means to specify how digital documents should be printed onto physical paper. As seen in Figure 1, PDFs store not units of text—headings, paragraphs, or other meaningful prose elements—but single characters alongside their spacing, placement, and any metadata used for visual rendering on a page. As more and more documents became digital, users have relied this file format to create trillions of documents (PDF Association staff, 2015); yet, these documents remain difficult to leverage in LM pipelines because PDFs lack basic structure necessary for coherent prose, such as ground truth reading order.

```
Character: 'o'
Transform Matrix: (1.02, 0.0, 0, 1, 70.866, 709.481)
Font: JURTWD+Manrope-Bold, Size: 24.78710000000001
------------------------------
Character: 'l'
Transform Matrix: (1.02, 0.0, 0, 1, 86.490796356, 709.481)
Font: JURTWD+Manrope-Bold, Size: 24.78710000000001
------------------------------
Character: 'm'
Transform Matrix: (1.02, 0.0, 0, 1, 93.56999211600001, 709.481)
Font: JURTWD+Manrope-Bold, Size: 24.78710000000001
------------------------------
Character: 'O'
Transform Matrix: (1.02, 0.0, 0, 1, 116.299267074, 709.481)
Font: JURTWD+Manrope-Bold, Size: 24.78710000000001
------------------------------
Character: 'C'
Transform Matrix: (1.02, 0.0, 0, 1, 135.236115732, 709.481)
Font: JURTWD+Manrope-Bold, Size: 24.78710000000001
------------------------------
Character: 'R'
Transform Matrix: (1.02, 0.0, 0, 1, 153.894853128, 709.481)
Font: JURTWD+Manrope-Bold, Size: 24.78710000000001
```

Figure 1 Example of how PDFs represent textual content, such as this paper title, as individual glyphs with metadata.

Faithful content extraction and representation of digitized print documents has long been of interest, with early research efforts in the 1950s, and first commercial optical character recognition (OCR) tools debuting in the late 1970s (Mori et al., 1992). The release of Tesseract in 2006 represented a significant milestone, as the first high-quality, open-source OCR toolkit (Smith, 2013). The current landscape of PDF extraction toolkits

can be partitioned in pipeline-based systems and end-to-end models. Pipeline-based systems (MinerU, Wang et al. 2024a; Marker, Paruchuri 2025) are comprised of multiple ML components (*e.g.*, section segmentation, table parsing) chained together; some, such as Grobid (GRO, 2008–2025), VILA (Shen et al., 2022), and PaperMage (Lo et al., 2023), are tailored to scientific papers. On the other hand, end-to-end models parse a document with a single model. For example, Nougat (Blecher et al., 2023) and GOT Theory 2.0 (Wei et al., 2024) take images of PDF pages as input, and return plain text. Notably, while pipeline-based systems have historically focused on simply faithful extraction, end-to-end-systems have also made strides to enable *linearization* of this content—prescribing a flattening of this content to adhere to logical reading order—which can be quite challenging for layout-rich documents with many floating elements (e.g. multi-column documents with floating diagrams, headers, footnotes, and more). Recently, rapid advances in the proprietary LMs have led to significant improvements in end-to-end text extraction capabilities (Bai et al., 2025; Google, 2025). However, this capability comes at a steep price: for example, converting a million pages using GPT-4o can cost over $6,200 USD.[1]

We introduce **olmOCR**, a general-purpose context extraction and linearization toolkit to convert PDFs or images of documents into clean plain text:

- olmOCR is capable of processing a diversity of document types, covering different domains as well as visual layouts. It uses Markdown (Gruber, 2004) to represent structured content, such as sections, lists, equations and tables.
- Unlike other end-to-end models, olmOCR uses *both* text and visual information to obtain an accurate text representation of a documents. We develop document-anchoring, a technique to extract text and layout information from born-digital PDF documents. document-anchoring can be used to prompt VLMs alongside images of document pages to significantly improve extraction.
- To build olmOCR, we curate `olmOCR-mix-0225`, a dataset of nearly 260,000 PDF pages from a diverse set of PDFs crawled from the web and public domain books. This corpus is used to fine-tune `olmOCR-7B-0225-preview` from Qwen2-VL-7B-Instruct (Wang et al., 2024b). We release `olmOCR-mix-0225` to facilitate further research in document extraction, and open source model weights and code as part of our toolkit.
- olmOCR is a fully optimized pipeline compatible with both SGLang (Zheng et al., 2024) and vLLM (Kwon et al., 2023) inference engines. In our tests, we have been able to scale it efficiently from one to hundreds of GPUs. It achieves an amortized cost of less than $190 per million pages converted, or $1/32^{nd}$ of the price of calling GPT-4o APIs. Furthermore, olmOCR is resilient: it includes several heuristics to handle common parsing failures and metadata errors.

2 Methodology

Approach Many end-to-end OCR models, such as GOT Theory 2.0 (Wei et al., 2024) and Nougat (Blecher et al., 2023), exclusively rely on rasterized pages to convert documents to plain text; that is, they process images of the document pages as input to autoregressively decode text tokens. This approach, while offering great compatibility with image-only digitization pipelines, misses the fact that most PDFs are born-digital documents, thus already contain either digitized text or other metadata that would help in correctly linearizing the content.

In contrast, the olmOCR pipeline leverages document text and metadata. We call this approach **document-anchoring**. Figure 2 provides an overview of our method; document-anchoring extracts coordinates of salient elements in each page (*e.g.*, text blocks and images) and injects them alongside raw text extracted from the PDF binary file. Crucially, the anchored text is provide as input to any VLM *alongside* a rasterized image of the page.

Our approach increases the quality of our content extraction. We apply document-anchoring when prompting GPT-4o to collect silver training samples, when fine-tuning `olmOCR-7B-0225-preview`, and when performing inference with the olmOCR toolkit.

[1] With batch pricing, at $1.25 USD (input) and $5.00 USD (output) per million tokens in Feb 2025.