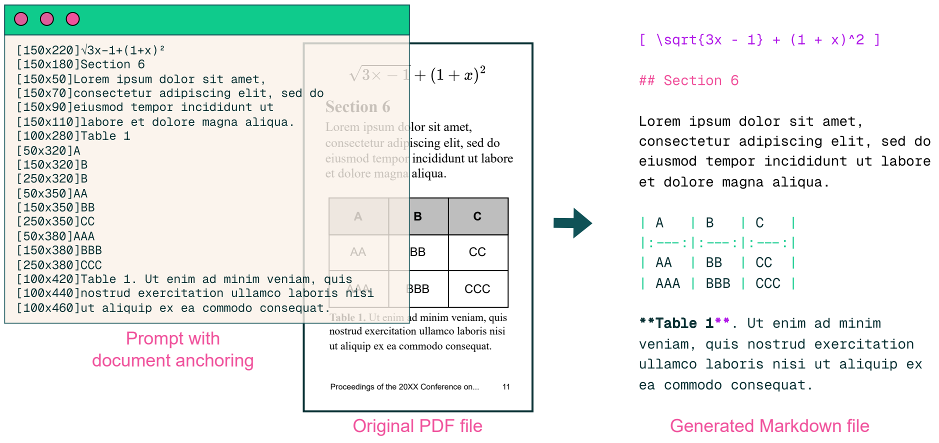
olmOCR: Unlocking Trillions of Tokens in PDFs with Vision Language Models

Jake Poznanski♥ Aman Rangapur

Jon Borchardt Jason Dunkelberger Regan Huff Daniel Lin Christopher Wilhelm

Kyle Lo♥ Luca Soldaini♥

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



```
[150x220]√3x-1+(1+x)²
[150x180]Section 6
[150x50]Lorem ipsum dolor sit amet,
[150x70]consectetur adipiscing elit, sed do
[150x90]eiusmod tempor incididunt ut
[150x110]labore et dolore magna aliqua.
[100x280]Table 1
[50x320]A
[150x320]B
[250x320]B
[50x350]AA
[150x350]BB
[250x350]CC
[50x380]AAA
[150x380]BBB
[250x380]CCC
[100x420]Table 1. Ut enim ad minim veniam, quis
[100x440]nostrud exercitation ullamco laboris nisi
[100x460]ut aliquip ex ea commodo consequat.
```

$$\sqrt{3\times - 1} + (1+x)^2$$

Section 6

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

A	B	C
AA	BB	CC
AAA	BBB	CCC

Table 1. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Proceedings of the 20XX Conference on... 11

```
[ \sqrt{3x - 1} + (1 + x)^2 ]

## Section 6

Lorem ipsum dolor sit amet,
consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore
et dolore magna aliqua.

| A   | B   | C   |
|:---:|:---:|:---:|
| AA  | BB  | CC  |
| AAA | BBB | CCC |

**Table 1**. Ut enim ad minim
veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex
ea commodo consequat.
```

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. Traditional open source tools often produce lower quality extractions compared to vision language models (VLMs), but reliance on the best VLMs can be prohibitively costly (e.g., over $6,240 USD per million PDF pages for GPT-4o) or infeasible if the PDFs cannot be sent to proprietary APIs. We present OLMOCR, an open-source 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 `olmOCR-mix-0225`, 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 can convert a million PDF pages for only $176 USD. To aid comparison with existing systems, we also introduce OLMOCR-BENCH, a curated set of 1,400 PDFs capturing many content types that remain challenging even for the best tools and VLMs, including formulas, tables, tiny fonts, old scans, and more. We find OLMOCR outperforms even top VLMs including GPT-4o, Gemini Flash 2 and Qwen-2.5-VL. We openly release all components of OLMOCR: our fine-tuned VLM model, training code and data, an efficient inference pipeline that supports vLLM and SGLang backends, and benchmark OLMOCR-BENCH.

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

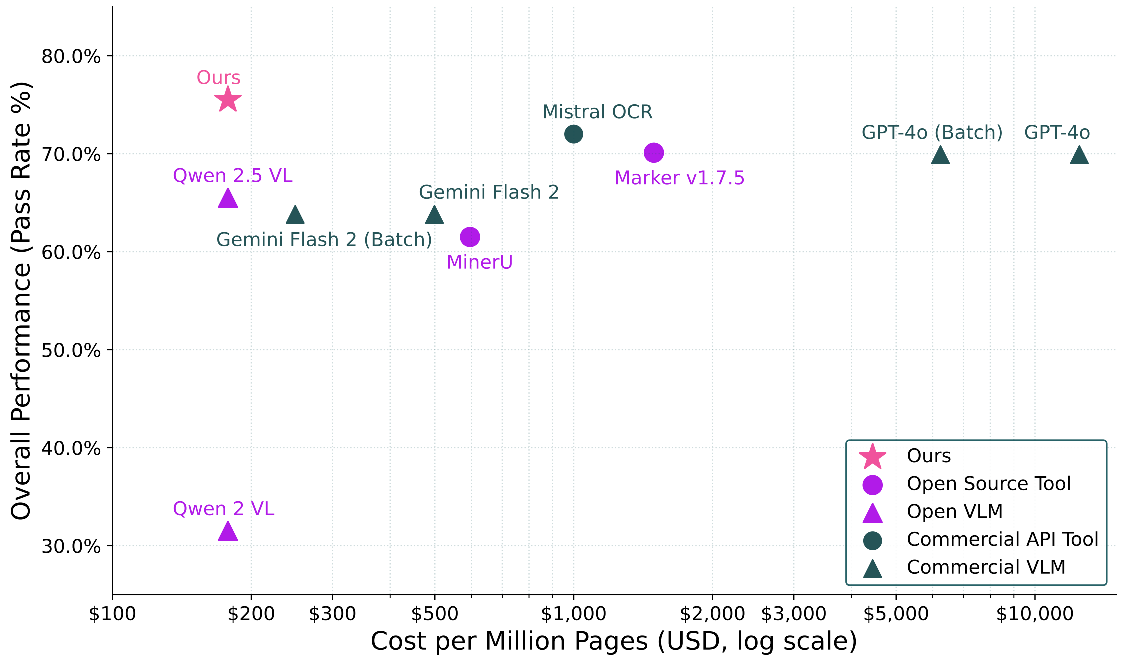


Figure 1 Performance-to-cost of olmOCR compared to a range of methods for PDF linearization and content extraction. Baselines include open and closed-source **specialized tools** and **general VLMs** prompted to perform this task. Performance is calculated on olmOCR-Bench, while Cost is calculated using commercial API pricing or the L40S GPU hourly rate (full details in Appendix B). As olmOCR uses a fine-tuned Qwen 2 VL model (7B), they share the same inference cost; performance differences are a result of fine-tuning on our dataset `olmOCR-mix-0225`.

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 2, 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.

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

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

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., 2023a), 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 suitable for language model development. Our contributions in this work are as follows:

- **Data.** We create `olmOCR-mix-0225`, a collection of 260,000 crawled PDF pages paired with their OCR output by GPT-4o, that we use to train our models. These documents represent a diverse set of publicly available PDFs, with a skew towards academic papers, public domain books, legal documents, brochures, and more.
- **Benchmark.** We develop olmOCR-Bench, a comprehensive benchmark for evaluating document extraction tools. Unlike existing evaluation methods, olmOCR-Bench uses simple, natural binary rules, like software unit tests, that enable direct comparisons across different OCR systems without relying on fuzzy gold reference matching or LLM-as-judge for evaluation. The benchmark covers 1,400 PDF pages with over 7,000 unit-test cases spanning diverse document types.

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

- **Model and Code.** We fine-tune Qwen2-VL-7B-Instruct (Wang et al., 2024b) on `olmOCR-mix-0225`, producing `olmOCR-7B-0225-preview`. We package our VLM in the OLMOCR Python toolkit, written to scale efficiently from one to hundreds of GPUs using SGLang (Zheng et al., 2024) and vLLM (Kwon et al., 2023) inference engines. OLMOCR achieves state-of-the-art performance on our benchmark, even outperforming Qwen-2.5-VL-7B while remaining more cost-effective than existing alternatives, including commercial APIs; OLMOCR can produce high-quality plain text at less than $176 per million PDF pages.
- **Downstream Use.** We demonstrate real-world impact by applying OLMOCR to process the 7.9M original PDFs in peS2o (Soldaini and Lo, 2023), a widely-used corpus of linearized scientific articles used in language model pretraining. We show that training on the newly extracted data called OLMOCR-PES2O can improve language model pretraining, observable even in downstream benchmark performance.

2 Creating and Training on olmOCR mix

We face two challenges in data acquisition necessary for developing a VLM for our task: (1) acquiring a large, diverse set of PDFs and (2) obtaining their linearized plain text as supervision targets.

2.1 Crawling PDFs

Source	Unique docs	Total pages
Web crawled PDFs	96,929	240,940
Internet Archive books	5,896	17,701
Total	*102,825*	*258,641*

Table 1 `olmOCR-mix-0225` composition by source. Web crawled PDFs are sampled from a set of over 240 million documents crawled from public websites. Books in the Internet Archive set are in the public domain.

Document type	Fraction
Academic	55.9%
Brochure	11.2%
Legal	10.2%
Books	6.8%
Table	5.6%
Diagram	4.7%
Slideshow	1.9%
Other	3.7%

Table 2 `olmOCR-mix-0225` PDFs breakdown by document type. Estimated by sampling 707 pages, classified using `gpt-4o-2024-11-20`. Prompt in Appendix E.3.

We randomly sample PDFs from an internal dataset of 240 million PDFs crawled from public internet sites, as well as PDFs of public domain books sourced from the Internet Archive. While the web crawled set is often born-digital documents, PDFs from the Internet Archive consist of image scans. We then perform a set of filters: Using the Lingua package (Emond, 2025), we identify and filter out non-English documents. Further, we remove any document that failed to be parsed by `pypdf`, contains spam keywords, is a fillable form, or whose text is too short.[2] We then sampled (up to) three pages uniformly at random from each PDF. We summarize the data distribution in Tables 1 and 2.

2.2 Generating Linearized Plain Text

Obtaining supervision targets for converting PDF to plain text presents a fundamental challenge. First, human annotation is prohibitively expensive and can be error-prone. Second, existing tools that extract content from PDF internals don't work on document images, but also don't provide reliable ground truth due to extraction errors from brittle heuristics. In this work, we turned to data generation using GPT-4o to reliably convert PDF pages to linearized plain text.[3]

[2] An implementation of these heuristics is available on GitHub: `/olmocr/filter/filter.py#L14-L112`.

[3] In October 2024, we evaluated several leading VLMs for data generation. Gemini 1.5 was eliminated due to frequent `RECITATION` errors (though this was resolved by February 2025), GPT-4o mini produced excessive hallucinations, and Claude Sonnet 3.5 was cost-prohibitive. We selected `gpt-4o-2024-08-06` as it offered the optimal balance of accuracy, reliability, and cost-efficiency in batch mode.

Yet, GPT-4o does not produce sufficiently high-fidelity plain text on its own; for high-density pages or complex layouts, we found it is prone to omitting content, rewriting or completing content in a manner unfaithful to the original, or captioning images when not instructed to do so. To help guide GPT-4o generations, we experiment with augmenting the visual input (PDF page raster) with text blocks and position information extracted from the page. As mentioned in A, we refer to this approach as DOCUMENT-ANCHORING.

We use the `pypdf` (PyPDF, 2012–2025) library to extract a representation of the page's structure from the PDF's internal data. We note that this representation is *highly noisy*: reading order is not preserved and main content is interwoven with boilerplate text and PDF rendering-related artifacts. We sample blocks from this long extraction to add to the prompt until maximum input length is exceeded; we prioritize text blocks and images which are located at the start and end of the document.

Finally, we instruct GPT-4o to respond with structured output to our requests. We report the full JSON schema in Appendix E.1. This forces the model to first extract page metadata, such as language, page orientation, and presence of tables, *before* generating the text of the page in a natural reading order. This format allows for more efficient processing of output; further, we found it crucial to ensure that GPT-4o does not generate captions of images when no text is present on the page. Overall, we find DOCUMENT-ANCHORING indeed improves the output quality of GPT-4o according to our benchmark (§3) reported in Table 4.

2.3 Model Training

Fine-tuning While DOCUMENT-ANCHORING could be used to prompt any language model, its performance may depend on the model (Table 4), making it best suited as a data generation technique. This leaves open the question whether a smaller, specialized VLM can be as accurate as optimized prompting of a larger, general-purpose model.

Starting from a Qwen2-VL-7B-Instruct checkpoint, we fine-tune `olmOCR-7B-0225-preview` on `olmOCR-mix-0225`. Training is implemented using Hugging Face's `transformers` library (Wolf et al., 2020). We use an effective batch size of 4, learning rate of 1e-6, AdamW optimizer, and a cosine annealing schedule for 10,000 steps (roughly 1.2 epochs).[4] We use single node with 8 x NVIDIA H100 (80GB) GPUs. A single training run took 16 node hours, with all training experiments totaling 365 node hours.

During fine-tuning, we slightly alter the DOCUMENT-ANCHORING prompt, removing some instructions and shrinking the image size so that PDF pages are rendered to a maximum dimension of 1024 pixels on the longest edge. The simplified text prompt is in Appendix E.2. The prompt is capped to 6,000 characters, so a typical prompt uses about 1,000 tokens to encode a page image, 1,800 tokens for the anchor text, for about 3,000 total input tokens. Each training example was truncated to 8,192 tokens to cover cases when the prompt was unusually large. Loss was masked so only the final response tokens participated in the loss calculation.

We keep the same structured JSON output that was present in the outputs of `olmOCR-mix-0225`. More training evaluations are noted in Appendix C.

3 Building olmOCR-Bench

We develop OLMOCR-BENCH to systematically evaluate PDF linearization and content extraction performance across diverse tools and models. OLMOCR-BENCH operates by assessing a series of predefined pass-or-fail "unit-tests"—*Given an input whole PDF, does the plain text output satisfy a specific property or contain a specific element?* Each test is designed to be simple, unambiguous, and deterministically machine-verifiable. This avoids reliance on model-based evaluators which can be biased towards favoring their own generations (Panickssery et al., 2024). It also avoids use of soft metric (e.g., edit distance, ROUGE) comparisons against reference text which might fail to reveal fine-grained yet semantically important content extraction errors, as is the case with incorrect math formulas (e.g., x^i vs x_i). OLMOCR-BENCH comprises 1,402 distinct PDF documents derived from diverse source repositories, covered by 7,010 unique test cases. Some test patterns apply to any document type (e.g., presence, absence, reading order) while others are motivated by particular challenging yet important content extraction targets (e.g., tables, math formulas); see Table 3 for a breakdown.

[4]We manually tuned hyperparameters alongside other data curation decisions (e.g. DOCUMENT-ANCHORING prompt) throughout development. To support this iterative cycle, we relied on manual side-by-side evaluation as shown in Appendix Figure 6.

	Presence	Absence	Read Order	Table	Formula	Total tests
arXiv Math (AM)	-	-	-	-	2,927	2,927
Old Scans Math (OSM)	-	-	-	-	458	458
Tables (TA)	-	-	-	1,020	-	1,020
Old Scans (OS)	279	70	177	-	-	526
Headers Footers (HF)	-	753	-	-	-	753
Multi Column (MC)	-	-	884	-	-	884
Long Tiny Text (LTT)	442	-	-	-	-	442
Total Tests	721	823	1,061	1,020	3,385	7,010

Table 3 Counts of unit test types in OLMOCR-BENCH.

3.1 Unit Test Categories

We designed 5 distinct test categories, each designed to assess specific aspects of linearization and context extraction performance. We describe the test definitions and scoring methods below:

- **Text Presence**: Verifies that a text segment (typically spanning 1-3 sentences) is correctly identified within the plain text output. Soft/fuzzy matching is allowed, as well as specifying if the text must be in the first N or last N characters of the document. Case-sensitive by default.
- **Text Absence**: Verifies that a text segment is successfully excluded from the plain text output. This category primarily targets peripheral content such as recurring headers, footers, and pagination markers. Soft/fuzzy matching is allowed, as well as specifying if the text must be in the first N or last N characters of the document. *Not* case-sensitive by default.
- **Natural Reading Order**: Verifies the order between two text segments. For instance, on a PDF with multiple news articles on one page, we can test for whether the first sentence of the first article appears after the heading of that article; yet such tests can be designed to not penalize for the order of the articles themselves. Soft matching is allowed, case-sensitive by default.
- **Table Accuracy**: Checks that the plain text output contains a table with a cell with a given value, and that its neighboring cells have certain properties. For instance, one can validate this page has a table with a cell containing "4.5%" and above that is a cell containing "2.4%". Both Markdown and HTML based tables are supported, though many cases depend on `rowspan` and `colspan` information being preserved, which is possible only in HTML based tables.
- **Math Formula Accuracy**: Checks that the plain text output contains a given math equation. We render a reference LaTeX equation using KaTeX in a headless browser and extract all rendered symbols and their (visual) bounding boxes. Then we check if a matching collection of symbols, with the same relative orientations, exists anywhere in the final OCR document. For instance, if searching for $f(x) = \int_{-3}^{3} x^2 dx$ on a page, we look for an equation where $\int$ appears to the left of a x, x appears to the left of dx, 3 appears above -3, and so on. This is similar to the method described by Wang et al. (2025), but ours is simpler due to the test being Pass/Fail only.
- **Baseline**: Each PDF document by default also receives a baseline or default test case, which checks that some plain text output containing alphanumeric characters was actually produced for that page, that such output does not have a string of repeating N grams at the end (longer than 30), and that the output does not contain any characters from the Chinese, Japanese, or Emoji Unicode charsets.[5]

In all cases where text is compared, we perform basic string normalization, such as converting $<br>$s to newlines, normalizing all whitespace to single ASCII spaces, removing Markdown bold/italics, normalizing quotes/hyphens to ASCII, and converting all Unicode to NFC format.

[5]This is to test for common failure cases of models, such as accidentally switching languages and generating repeated outputs. The handful of pages which do legitimately contain such charsets are manually flagged and excluded from such test conditions.

3.2 Sourcing Documents and Creating Tests

We define 7 distinct document types that we found olmOCR (or its earlier iterations) often struggled to process and defined custom acquisition strategies for each (described below). We removed documents that both contained PII and were not meant for public dissemination; prompt in Appendix F.2.2. We also decontaminate against documents that appear in `olmOCR-mix-0225` via URL level deduplication. To scale creation of test cases over these documents, we combined manual design and review with prompting GPT-4o; further details and prompts are in Appendix F. Visualize sample documents in Appendix F.3.

- **arXiv Math (AR)** We downloaded a recent set of papers from the math subset of arXiv, selecting manuscripts with a single TeX source file and corresponding rendered PDF. To select a candidate LaTeX expression from a page to use in a test, we (1) ran olmOCR to identify candidate pages with TeX, (2) match pages back to original TeX source, and (3) validate matched TeX rendering compatibility with KaTeX.

  We manually verify the final set of test cases to exclude instances where custom macros produce renderings that deviate from standard LaTeX and to split multi-part equations into smaller test cases.
- **Old Scans Math (OSM)** We crawl old, public domain math textbooks from the Internet Archive[6], extracting random pages from these documents. We similarly use olmOCR to find candidate pages with formulas, but this time manually annotate each formula on the page to use as test cases.
- **Tables (TA)** We sampled more documents from the same internal crawled PDF repository used to create `olmOCR-mix-0225` and filtered to those which had tables using a simple prompt with Gemini-Flash-2.0. On pages with tables, we prompted Gemini-Flash-2.0 for the relationships between randomly chosen cells. We manually reviewed those tests for accuracy.
- **Old Scans (OS)** We sampled historical letters and typewritten documents with existing human transcriptions from the Library of Congress[7] digital archives. We then wrote a small script to generate Natural Reading Order cases consisting of sentences that were naturally before or after one another in the original human transcriptions. We manually added test cases to cover some headers/footers which should have been excluded from any OCR version of these documents. All of the test cases then underwent a second pass of human review for accuracy.
- **Headers Footers (HF)** We sampled documents from the same internally crawled PDF repository as `olmOCR-mix-0225`. We used DocLayout-YOLO (Zhao et al., 2024) to identify page regions labeled as headers or footers using the `abandon` category. To extract the text from these header/footer regions, we visually mask out the rest of the document and prompt Gemini-Flash-2.0 for the content. These extracted snippets are added as test cases that should be absent in linearized output. We manually reviewed to remove mistakenly filtered text and to set conditions such as limiting the search area to the first N or last N characters. For example, if a page number "5" appears at the bottom of a page, we test to ensure that output plain text does not contain "5" in the last 20 characters, but still allow for a "5" that may appear earlier in the text.
- **Multi Column (MC)** We visually sample documents from our internal crawled PDF repository to find documents with multi-column layouts and multiple articles on one page. We use Claude-Sonnet-3.7 to render those pages to HTML, and from that HTML, we extract text segments before/after one another. We manually review each entry for accuracy. We purposely select simple text blocks from coherent regions of the document, and avoid including any math formulas, superscripts, or subscripts in these tests.
- **Long Tiny Text (LTT)** We crawled documents from the Internet Archive containing a large amount of dense, small print on a single page. Such documents include pages from a dictionary or pages of references from academic papers. We then generate test cases using Gemini-Flash-2.0 and verify them manually.

3.3 Scoring

We run each of the PDF pages across each of our tools and methods to produce a markdown or plain text document. As all tests are Pass/Fail, we simply report percentage of tests passed, macro-averaged by document type. We evaluate each of the tests to get a percentage correct score for each test source (plus the default baseline tests). The final score for each tool is the average of the percentage across each test source. This

[6]https://archive.org
[7]https://crowd.loc.gov

captures the difficulty we faced at times of finding and validating enough cases from each source, but we roughly feel that each source represents an important capability for an OCR system to have.

$$\text{Overall score} = \frac{1}{N} \sum_{s \in \text{Document sources}} \text{Score}(s)$$

4 Evaluating olmOCR

First, we evaluate olmOCR on olmOCR-Bench against a range of linearization tools and VLMs (Section §4.1). We then quantify the usefulness olmOCR for language modeling by continued pretraining on an OLMo 2 checkpoint (OLMo et al., 2024) on content extracted and linearized with our toolkit (Section §4.2).

Additional evaluations, studying how faithful olmOCR is to its teacher model (Section §C.1), and pairwise ELO comparison (Section §C.2) are available in the appendix.

4.1 olmOCR-Bench Results

From Table 4, we see that olmOCR significantly outperforms both the best commercial dedicated OCR tool (Mistral) as well as both GPT-4o, its teacher model, and Qwen 2.5 VL, which is an update to Qwen 2 VL, which was the base model for `olmOCR-7B-0225-preview`. We note that we developed olmOCR-Bench *after* training `olmOCR-7B-0225-preview` to prevent unfairly iterating on the benchmark before comparing with other methods. Qualitatively, olmOCR produces significantly cleaner plain text than specialized open-source tools (visualized in Appendix G).

Model	AR	OSM	TA	OS	HF	MC	LTT	Base	Overall
GOT OCR	52.7	52.0	0.2	22.1	93.6	42.0	29.9	94.0	48.3 ± 1.1
Marker v1.7.5	76.0	57.9	57.6	27.8	84.9	72.9	84.6	99.1	70.1 ± 1.1
MinerU v1.3.10	75.4	47.4	60.9	17.3	**96.6**	59.0	39.1	96.6	61.5 ± 1.1
Mistral OCR API	**77.2**	67.5	60.6	29.3	93.6	71.3	77.1	**99.4**	72.0 ± 1.1
GPT-4o (No Anchor)	51.5	**75.5**	69.1	40.9	94.2	68.9	54.1	96.7	68.9 ± 1.1
GPT-4o (Anchored)	53.5	74.5	70.0	40.7	93.8	69.3	60.6	96.8	69.9 ± 1.1
Gemini Flash 2 (No Anchor)	32.1	56.3	61.4	27.8	48.0	58.7	**84.4**	94.0	57.8 ± 1.1
Gemini Flash 2 (Anchored)	54.5	56.1	**72.1**	34.2	64.7	61.5	71.5	95.6	63.8 ± 1.2
Qwen 2 VL (No Anchor)	19.7	31.7	24.2	17.1	88.9	8.3	6.8	55.5	31.5 ± 0.9
Qwen 2.5 VL (No Anchor)	63.1	65.7	67.3	38.6	73.6	68.3	49.1	98.3	65.5 ± 1.2
Ours (v0.1.75 No Anchor)	71.5	71.4	71.4	**42.8**	94.1	77.7	71.0	97.8	74.7 ± 1.1
Ours (v0.1.75 Anchored)	74.9	71.2	71.0	42.2	94.5	**78.3**	73.3	98.3	**75.5 ± 1.0**

Table 4 Evaluation results on olmOCR-Bench grouped by document types. Best unit test pass rate in each column is bold. 95% CI calculated by bootstrapping with 10k samples.

4.2 Downstream Evaluation

To assess the impact of improved PDF linearization, we experiment using an intermediate checkpoint of `OLMo-2-1124-7B` and continued pretraining using content extracted from a fixed collection of PDFs but with different linearization tools. This ablation procedure has been used to assess data quality in (Blakeney et al., 2024; Grattafiori et al., 2024; OLMo et al., 2024).

For our baseline, we use PDF extracted tokens from `peS2o` (Soldaini and Lo, 2023), 58B tokens from academic papers derived using Grobid (gro, 2008–2025) from the S2ORC (Lo et al., 2020) paper collection and further cleaned with heuristics for language modeling. To represent olmOCR, we identify the same documents used in peS2o, acquire their source PDFs from the upstream S2ORC pipeline, and reprocess them using olmOCR. For these two versions of peS2o, we train the 7B checkpoint for another 50B tokens. As shown in Table 5, replacing the original peS2o tokens extracted via `Grobid + rules` with those processed used olmOCR results in a +1.3 percentage point average improvement on widely-reported LM benchmark tasks, including MMLU (Hendrycks et al., 2021), ARC$_C$, DROP (Dua et al., 2019), HellaSwag (Zellers et al., 2019), NaturalQuestions (Kwiatkowski et al., 2019), WinoGrande (Sakaguchi et al., 2019).

peS2o version	Average	MMLU	ARC$_C$	DROP	HSwag	NQ	WinoG
Grobid + rules (Soldaini and Lo, 2023)	53.9	**61.1**	75.0	42.3	57.4	**29.4**	**58.3**
olmOCR-peS2o	**55.2**	**61.1**	**76.4**	**43.7**	**62.6**	29.1	58.0

Table 5 Comparison on OLMo 2 (OLMo et al., 2024) downstream evaluation tasks of `OLMo-2-7B-1124` on 50B of original peS2o tokens vs 50B tokens from the same source PDFs but processed with olmOCR.

4.3 Cost Evaluation

Finally, when considering real-world use, cost efficiency is just as important as performance. We present a summary of inference costs in Table 6. To contextualize the value of olmOCR, at 1,000 tokens per page, to process all of peS2o PDFs can already cost $10.3M in H100 usage. In comparison, Mistral OCR is a commercial API tool specializing in this task, yet is over five times more expensive, making it even more prohibitive to use for language modeling. See Appendix B for details on pricing and cost calculations.

Model	Hardware	Tokens/sec	Pages/USD	Cost per million pages
GPT-4o	API	-	80	$12,480
	Batch	-	160	$6,240
Marker v1.7.5 (Force OCR)	H100	332	674	$1484
Mistral OCR	API	-	1,000	$1,000
MinerU	L40S	238	1,678	$596
Gemini Flash 2	API	-	2,004	$499
	Batch	-	4,008	$249
olmOCR	L40S	906	**5,697**	**$176**
	H100	3,050	**5,632**	**$178**

Table 6 Inference cost comparison against other OCR methods. NVIDIA L40S estimated at $0.79 per hour, H100 80GB estimated at $2.69 per hour. We measured a 12% retry rate for olmOCR. Full cost breakdown in Appendix B.

5 Related Work

Tools and Models for Linearizing PDFs to Plain Text. Many tools have existed for this task, some are parsers of born-digital PDF internals while others are OCR tools on top of image rasters of PDF pages. As machine learning matured, more people started developing models that automate this PDF parsing; examples include LayoutLM (Xu et al., 2020) and VILA (Lin et al., 2024). Tools have been developed around use of these models, including PaperMage (Lo et al., 2023b), or have updated to include their own custom trained models, like Grobid (gro, 2008–2025). Commercial API providers began integrating VLMs with document processing capabilities: OpenAI introduced GPT-4 Vision (OpenAI et al., 2024) in September 2023, Google launched Gemini (Gemini Team, 2025) in December 2023 with significant enhancements throughout 2024 as multimodal models became more powerful and accessible. Despite these developments, there remained a notable absence that specifically train a VLM for this task and package this capability into comprehensive, production-ready software libraries. Our work addresses this gap, developing alongside concurrent efforts such as Mistral (Mistral, 2025), Qwen VL (Bai et al., 2023) which we systematically evaluate against.

Benchmarking VLMs on Linearization. Several benchmarks have been developed for evaluating document understanding and content extraction. Established datasets such as FUNSD (Guillaume Jaume, 2019) focus on form understanding with typewritten content, SROIE (Huang et al., 2019) concentrates on information extraction from scanned receipts, and RVL-CDIP (Harley et al., 2015) contains scanned documents. However, these datasets exhibit significant limitations: they are predominantly domain specific, targeting narrow document categories with constraint formatting variations, while our approach leverages a diverse corpus spanning multiple domains and document types. Additionally, traditional benchmarks often focus on isolated extraction tasks (e.g., exclusively evaluating tables with PubTabNet (Zhong et al., 2020), or mathematical formulas with specialized detection frameworks (Zhong et al., 2021)), whereas our benchmark evaluates performance across a comprehensive spectrum of extraction challenges. Further, their evaluation method is brittle, typically relying on exact string matching against predefined gold-standard tokens (which makes it difficult to compare methods that produce different tokenizations). In contrast, our unit-test-style evaluation framework enables equitable assessment across diverse implementations regardless of their underlying tokenization mechanisms, providing a more generalizable evaluation paradigm for document understanding systems.

Linearization for Language Modeling. Significant work has been done on how to curate data for language modeling, with significant efforts on topics like data filtering (Soldaini et al., 2024; Penedo et al., 2024; Li et al., 2024) and source mixing (Wettig et al., 2025; Liu et al., 2024). However, relatively little attention has been directed toward understanding the impact of linearization processes on downstream model training. DCLM (Li et al., 2024) and RefinedWeb (Penedo et al., 2023) touched on some aspects of this challenge, utilizing tools like Resiliparse (Bevendorff et al., 2018) and Trafilatura (Barbaresi, 2021), but their approaches were restricted to web based textual content. OpenWebMath (Paster et al., 2023) showed accurate content extraction is important for specialized domains, like mathematical formulas. For PDF-based content specifically, similar research contributions remain limited, which motivates this work.

6 Conclusion

We introduce olmOCR, an open-source toolkit for converting PDF documents into clean plain text. Our approach combines document-anchoring, a novel prompting technique that leverages available metadata in born-digital PDFs, with a fine-tuned 7B parameter vision language model to achieve results competitive with closed commercial solutions at a fraction of the cost. We openly release our training set `olmOCR-mix-0225` to enable others to further develop their own VLMs.

To rigorously evaluate the system, we developed olmOCR-Bench, a benchmark of 7,010 test instances across 1,403 PDFs. It includes Pass/Fail unit tests for text presence, reading order, tables, formulas, and baseline functionality. The documents span categories from scientific papers to historical manuscripts, enabling robust assessment across diverse linearization and context extraction challenges.

Our released efficient inference pipeline contains everything needed to start converting anything from single documents to million-page archives of PDFs. We hope olmOCR's ability to efficiently process millions of documents will help unlock new sources of training data for language models, particularly from high-quality PDF documents that are currently underrepresented in existing datasets that rely heavily solely on crawled web pages.

Acknowledgments

This work would not be possible without the support of our colleagues at Ai2. We thank Byron Bischoff, Aaron Sarnat, Huy Tran, Sam Skjonsberg, Eric Marsh, and Chris Newell for help setting up the live demo; Taira Anderson, Sruthi Sreeram for program management support; Will Smith and Crystal Nam for legal guidance; Michael Schmitz, Caitlin Wittlif and Carissa Schoenick for various indirect support. We also thank Benjamin Charles Germain Lee for helpful feedback and suggestions on evaluation and potential use cases.

References

GROBID. https://github.com/kermitt2/grobid, 2008–2025.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond, 2023. https://arxiv.org/abs/2308.12966.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv [cs.CV]*, February 2025.

Adrien Barbaresi. Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 122–131. Association for Computational Linguistics, 2021. https://aclanthology.org/2021.acl-demo.15.

Janek Bevendorff, Benno Stein, Matthias Hagen, and Martin Potthast. Elastic ChatNoir: Search Engine for the ClueWeb and the Common Crawl. In Leif Azzopardi, Allan Hanbury, Gabriella Pasi, and Benjamin Piwowarski, editors, *Advances in Information Retrieval. 40th European Conference on IR Research (ECIR 2018)*, Lecture Notes in Computer Science, Berlin Heidelberg New York, March 2018. Springer.

Cody Blakeney, Mansheej Paul, Brett W. Larsen, Sean Owen, and Jonathan Frankle. Does your data spark joy? performance gains from domain upsampling at the end of training, 2024. https://arxiv.org/abs/2406.03476.

Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural optical understanding for academic documents, 2023. https://arxiv.org/abs/2308.13418.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019. https://arxiv.org/abs/1903.00161.

Patrick Emond. Lingua-py: Natural language detection for python, 2025. https://github.com/pemistahl/lingua-py. Accessed: 2025-01-06.

Gemini Team. Gemini: A family of highly capable multimodal models, 2025. https://arxiv.org/abs/2312.11805.

Google. Explore document processing capabilities with the gemini API. https://web.archive.org/web/20250224064040/https://ai.google.dev/gemini-api/docs/document-processing?lang=python, 2025. Accessed: 2025-2-23.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathurx, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The Llama 3 herd of models, 2024. https://arxiv.org/abs/2407.21783.

Jean-Philippe Thiran Guillaume Jaume, Hazim Kemal Ekenel. Funsd: A dataset for form understanding in noisy scanned documents. In *Accepted to ICDAR-OST*, 2019.

Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2015.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. https://arxiv.org/abs/2009.03300.

Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar. Icdar2019 competition on scanned receipt ocr and information extraction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1516–1520. IEEE, 2019.

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*, 2021. `https://api.semanticscholar.org/CorpusID:250924870`.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. `https://aclanthology.org/Q19-1026/`.

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

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024.

Ji Lin, Hongxu Yin, Wei Ping, Yao Lu, Pavlo Molchanov, Andrew Tao, Huizi Mao, Jan Kautz, Mohammad Shoeybi, and Song Han. Vila: On pre-training for visual language models, 2024. `https://arxiv.org/abs/2312.07533`.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. *arXiv preprint arXiv:2407.01492*, 2024.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. S2ORC: The semantic scholar open research corpus. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.447. `https://aclanthology.org/2020.acl-main.447/`.

Kyle Lo, Zejiang Shen, Benjamin Newman, Joseph Chang, Russell Authur, Erin Bransom, Stefan Candra, Yoganand Chandrasekhar, Regan Huff, Bailey Kuehl, Amanpreet Singh, Chris Wilhelm, Angele Zamarron, Marti A. Hearst, Daniel Weld, Doug Downey, and Luca Soldaini. PaperMage: A unified toolkit for processing, representing, and manipulating visually-rich scientific documents. In Yansong Feng and Els Lefever, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 495–507, Singapore, December 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-demo.45. `https://aclanthology.org/2023.emnlp-demo.45/`.

Kyle Lo, Zejiang Shen, Benjamin Newman, Joseph Z Chang, Russell Authur, Erin Bransom, Stefan Candra, Yoganand Chandrasekhar, Regan Huff, Bailey Kuehl, et al. Papermage: a unified toolkit for processing, representing, and manipulating visually-rich scientific documents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 495–507, 2023b.

Kyle Lo, Joseph Chee Chang, Andrew Head, Jonathan Bragg, Amy X. Zhang, Cassidy Trier, Chloe Anastasiades, Tal August, Russell Authur, Danielle Bragg, Erin Bransom, Isabel Cachola, Stefan Candra, Yoganand Chandrasekhar, Yen-Sung Chen, Evie Yu-Yen Cheng, Yvonne Chou, Doug Downey, Rob Evans, Raymond Fok, Fangzhou Hu, Regan Huff, Dongyeop Kang, Tae Soo Kim, Rodney Kinney, Aniket Kittur, Hyeonsu B. Kang, Egor Klevak, Bailey Kuehl, Michael J. Langan, Matt Latzke, Jaron Lochner, Kelsey MacMillan, Eric Marsh, Tyler Murray, Aakanksha Naik, Ngoc-Uyen Nguyen, Srishti Palani, Soya Park, Caroline Paulic, Napol Rachatasumrit, Smita Rao, Paul Sayre, Zejiang Shen, Pao Siangliulue, Luca Soldaini, Huy Tran, Madeleine van Zuylen, Lucy Lu Wang, Christopher Wilhelm, Caroline Wu, Jiangjiang Yang, Angele Zamarron, Marti A. Hearst, and Daniel S. Weld. The semantic reader project. *Commun. ACM*, 67(10):50–61, September 2024. ISSN 0001-0782. doi: 10.1145/3659096. `https://doi.org/10.1145/3659096`.

Mistral. Mistral ocr, 2025. `https://mistral.ai/news/mistral-ocr`.

S Mori, C Y Suen, and K Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE. Institute of Electrical and Electronics Engineers*, 80(7):1029–1058, July 1992. ISSN 0018-9219,1558-2256. doi: 10.1109/5.156468.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze

Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James Validad Miranda, Jacob Daniel Morrison, Tyler C. Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Chris Wilhelm, Michael Wilson, Luke S. Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hanna Hajishirzi. 2 OLMo 2 Furious. *arXiv preprint*, 2024. https://api.semanticscholar.org/CorpusID:275213098.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. https://arxiv.org/abs/2303.08774.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. Llm evaluators recognize and favor their own generations, 2024. https://arxiv.org/abs/2404.13076.

Vik Paruchuri. Marker: Convert pdf to markdown + json quickly with high accuracy, 2025. https://github.com/VikParuchuri/marker. Version 1.4.0.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text. *arXiv preprint arXiv:2310.06786*, 2023.

PDF Association staff. Pdf in 2016: Broader, deeper, richer. *PDF Association*, December 2015. https://pdfa.org/pdf-in-2016-broader-deeper-richer/.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra-Aimée Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM:

Outperforming curated corpora with web data, and web data only. *ArXiv*, abs/2306.01116, 2023. https://api.semanticscholar.org/CorpusID:259063761.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024.

PyPDF. Pypdf: A pure-python pdf library. https://github.com/py-pdf/pypdf, 2012–2025. Accessed: 2025-01-06.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019. https://arxiv.org/abs/1907.10641.

Zejiang Shen, Kyle Lo, Lucy Lu Wang, Bailey Kuehl, Daniel S. Weld, and Doug Downey. VILA: Improving structured content extraction from scientific PDFs using visual layout groups. *Transactions of the Association for Computational Linguistics*, 10:376–392, 2022. doi: 10.1162/tacl_a_00466. https://aclanthology.org/2022.tacl-1.22/.

Ray W Smith. History of the tesseract OCR engine: what worked and what didn't. In Richard Zanibbi and Bertrand Coüasnon, editors, *Document Recognition and Retrieval XX*, volume 8658, page 865802. SPIE, February 2013. doi: 10.1117/12.2010051.

Luca Soldaini and Kyle Lo. peS2o (Pretraining Efficiently on S2ORC) Dataset. Technical report, Allen Institute for AI, 2023. ODC-By, https://github.com/allenai/pes2o.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an open corpus of three trillion tokens for language model pretraining research, 2024. https://arxiv.org/abs/2402.00159.

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, and Conghui He. Mineru: An open-source solution for precise document content extraction, 2024a. https://arxiv.org/abs/2409.18839.

Bin Wang, Fan Wu, Linke Ouyang, Zhuangcheng Gu, Rui Zhang, Renqiu Xia, Bo Zhang, and Conghui He. Image over text: Transforming formula recognition evaluation with character detection matching, 2025. https://arxiv.org/abs/2409.03643.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution, 2024b. https://arxiv.org/abs/2409.12191.

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.

Alexander Wettig, Kyle Lo, Sewon Min, Hannaneh Hajishirzi, Danqi Chen, and Luca Soldaini. Organize the web: Constructing domains enhances pre-training data curation. *arXiv preprint arXiv:2502.10341*, 2025.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Perric Cistac, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*, KDD '20, page 1192–1200. ACM, August 2020. doi: 10.1145/3394486.3403172. http://dx.doi.org/10.1145/3394486.3403172.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. https://arxiv.org/abs/1905.07830.

Zhiyuan Zhao, Hengrui Kang, Bin Wang, and Conghui He. Doclayout-yolo: Enhancing document layout analysis through diverse synthetic data and global-to-local adaptive perception. *arXiv preprint arXiv:2410.12628*, 2024.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. Sglang: Efficient execution of structured language model programs, 2024. https://arxiv.org/abs/2312.07104.

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.

Yuxiang Zhong, Xianbiao Qi, Shanjun Li, Dengyi Gu, Yihao Chen, Peiyang Ning, and Rong Xiao. 1st place solution for icdar 2021 competition on mathematical formula detection, 2021. https://arxiv.org/abs/2107.05534.

A 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.

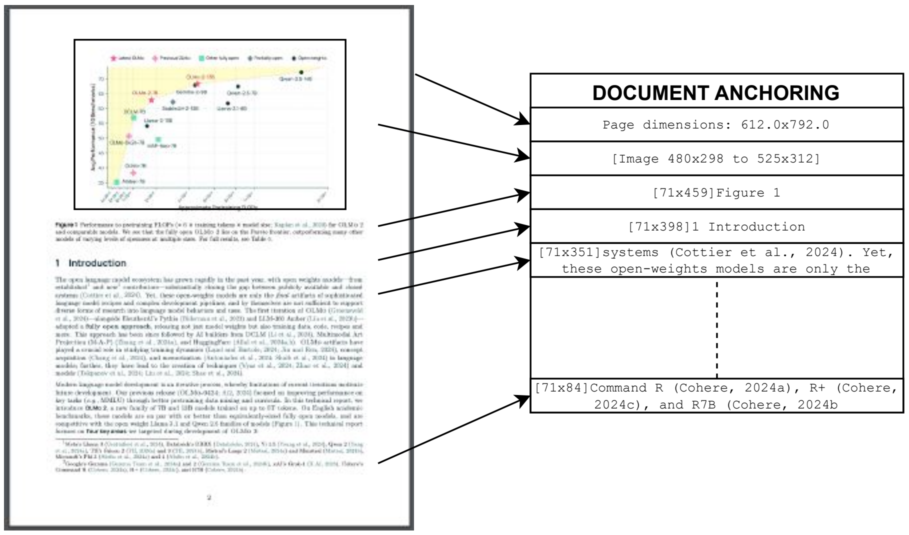

DOCUMENT ANCHORING
Page dimensions: 612.0x792.0
[Image 480x298 to 525x312]
[71x459]Figure 1
[71x398]1 Introduction
[71x351]systems (Cottier et al., 2024). Yet, these open-weights models are only the
[71x84]Command R (Cohere, 2024a), R+ (Cohere, 2024c), and R7B (Cohere, 2024b

Figure 3 Example of how DOCUMENT-ANCHORING works for a typical page. Relevant image locations and text blocks get extracted, concatenated, and inserted into the model prompt. When prompting a VLM for a plain text version of the document, the anchored text is used in conjunction with the rasterized image of a page.

In contrast, the OLMOCR pipeline leverages document text and metadata. We call this approach **DOCUMENT-ANCHORING**. Figure 3 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.

Implementation DOCUMENT-ANCHORING processes PDF document pages via the `pypdf` (PyPDF, 2012–2025) library to extract a representation of the page's structure from the underlying PDF. All of the text blocks and images in the page are extracted, including position information. Starting with the most relevant text blocks and images[8], these are sampled and added to the prompt of the VLM, up to a defined maximum character limit[9]. This extra information is then available to the model when processing the document.

Overall, we find that using prompts constructed using DOCUMENT-ANCHORING results in significantly fewer hallucinations. Prompting with just the page image was prone to models completing unfinished sentences, or to invent larger texts when the image data was ambiguous. Finally, while DOCUMENT-ANCHORING helps with quality on born-digital documents, our pipeline maintains high performance on documents that do not

[8]We prioritize text blocks and images which are located at the start and end of the document.

[9]We use a character limit for convenience and speed, but during training or inference, if a page's prompt exceeds the model's token limit, we just regenerate it with exponentially decreasing character limits until it is suitable.

have any digital metadata encoded in them. In these cases, the model will not have the benefit of seeing the internal structure of the PDF document, instead relying on just the rasterized image of a page to process the underlying document.

B Cost Estimates of PDF Extraction Systems

To estimate prices, we use rates provided by RunPod[10] as of February 2025. It prices a single on-demand NVIDIA L40S GPU at $0.79 USD per hour, and NVIDIA H100 80GB SXM at $2.69 USD per hour. Using these rates, costs (in USD) were computed as follows:

- **GPT-4o**: We evaluated GPT-4o in February 2025. We tested 1288 pages, which resulted in 3,093,315 input tokens at 833,599 output tokens. Priced at $2.50 per million input tokens and $10.00 per million output tokens, it resulted in a total of $16.07. Batch processing is priced at half of the cost, $8.03.
- **Mistral OCR**: As of May 2025, Mistral prices their OCR service at $1 per 1,000 pages, regardless of number of generated tokens.
- **MinerU**: We run the toolkit (version 1.3.10) on a single NVIDIA L40S GPU. It processed 1,288 pages in 58 minutes 22 seconds, costing $0.767.
- **Marker**: We run marker v1.7.5 using the marker command line with the `force_ocr` flag on 10,000 pages selected randomly from `olmOCR-mix-0225`. This took 5 hours, 31 minutes on an H100 node with 1 GPU, resulting in a price of $14.84 for 10,000 pages. [11]
- **Gemini Flash 2.0**: As of February 2025, it is priced $0.10 per 1 million input tokens, and $0.40 per 1 million output tokens. In our testing over the same 1,288 pages used to evaluate GPT-4o, it cost in $0.643.
- **olmOCR**: We tested the launch version of OLMOCR on both L40S and H100 GPUs. On L40s, it processed 1,288 test pages in 17 minutes, 10 seconds. The effective throughput of the model was 906 output tokens per second, plus a 12% reties rate. Overall, we estimate its costs at $0.226. On H100, OLMOCR generates 3,050 output tokens per second, resulting in a runtime of 5 minutes 7 seconds, for a cost of $0.229.

C Evaluation of Trained Models

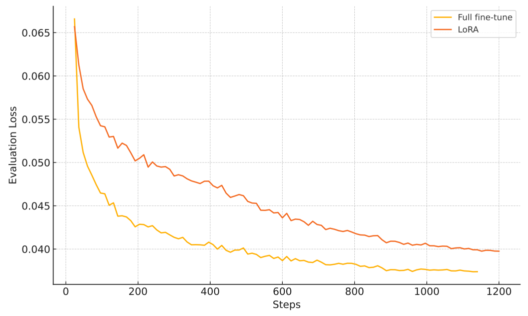


Figure 4 Validation Loss - Web PDFs

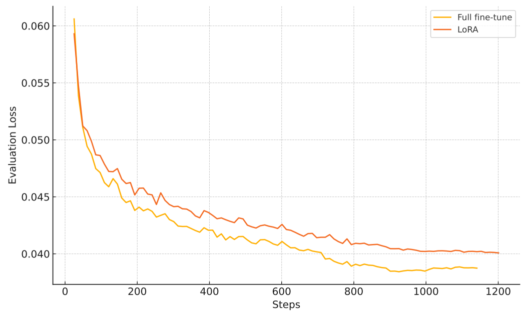


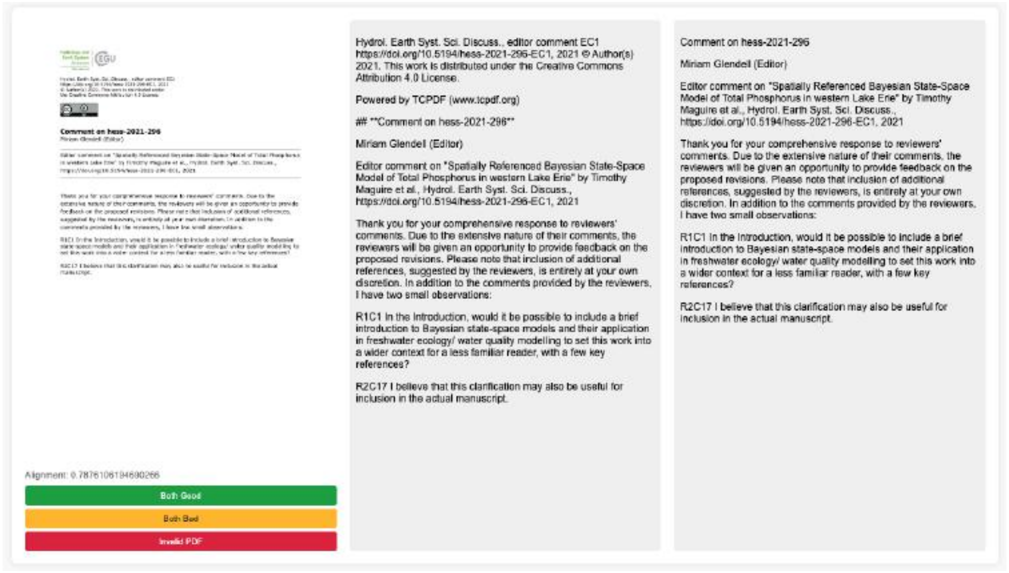
Figure 5 Validation Loss - Internet Archive Books

We track validation loss during training of `olmOCR-7B-0225-preview` against a development subset of `olmOCR-mix-0225` during fine-tuning; Figure 4 and Figure 5, show the loss curves for both the web PDFs and the Internet Archive books subsets. LoRA resulted in higher loss values compared to full fine-tuning, which we use for the final model.

To set hyperparameters and make other decisions during development, we relied on manual side-by-side evaluation as shown in Figure 6. A random selection of 20 to 50 documents were processed using two different

[10] https://www.runpod.io

[11] The force_ocr option is more expensive, but results in better performance. The Marker authors are working on improving performance on large GPUs with multiple workers which should lower this cost.


Hydrol. Earth Syst. Sci. Discuss., editor comment EC1 https://doi.org/10.5194/hess-2021-296-EC1, 2021 © Author(s) 2021. This work is distributed under the Creative Commons Attribution 4.0 License.

Powered by TCPDF (www.tcpdf.org)

Comment on hess-2021-296

Miriam Glendell (Editor)

Editor comment on "Spatially Referenced Bayesian State-Space Model of Total Phosphorus in western Lake Erie" by Timothy Maguire et al., Hydrol. Earth Syst. Sci. Discuss., https://doi.org/10.5194/hess-2021-296-EC1, 2021

Thank you for your comprehensive response to reviewers' comments. Due to the extensive nature of their comments, the reviewers will be given an opportunity to provide feedback on the proposed revisions. Please note that inclusion of additional references, suggested by the reviewers, is entirely at your own discretion. In addition to the comments provided by the reviewers, I have two small observations:

R1C1 In the Introduction, would it be possible to include a brief introduction to Bayesian state-space models and their application in freshwater ecology/ water quality modelling to set this work into a wider context for a less familiar reader, with a few key references?

R2C17 I believe that this clarification may also be useful for inclusion in the actual manuscript.

Comment on hess-2021-296

Miriam Glendell (Editor)

Editor comment on "Spatially Referenced Bayesian State-Space Model of Total Phosphorus in western Lake Erie" by Timothy Maguire et al., Hydrol. Earth Syst. Sci. Discuss., https://doi.org/10.5194/hess-2021-296-EC1, 2021

Thank you for your comprehensive response to reviewers' comments. Due to the extensive nature of their comments, the reviewers will be given an opportunity to provide feedback on the proposed revisions. Please note that inclusion of additional references, suggested by the reviewers, is entirely at your own discretion. In addition to the comments provided by the reviewers, I have two small observations:

R1C1 In the Introduction, would it be possible to include a brief introduction to Bayesian state-space models and their application in freshwater ecology/ water quality modelling to set this work into a wider context for a less familiar reader, with a few key references?

R2C17 I believe that this clarification may also be useful for inclusion in the actual manuscript.

Alignment: 0.7876106194690266

Figure 6 Example of side-by-side evaluation tool used during development. The software used to create these comparisons is released as open-source software as part of OLMOCR.

methods, and were displayed side by side along with the render of the document page. We also open source our evaluation tool to support qualitative inspection of this visually-rich data.

C.1 Alignment with Teacher Model

To compare the output of `olmOCR-7B-0225-preview` to the GPT-4o silver data in `olmOCR-mix-0225`, we build a document similarity metric which splits a document into words, uses Hirschberg's algorithm to align those words, and counts what proportion match.

We report alignment scores in Table 7. Overall, we find that `olmOCR-7B-0225-preview` has good alignment, 0.875 on average, with its teacher model. To calibrate this result, we also report GPT-4o self-alignment score of 0.954, which is simply from calling the model again; imperfect alignment here is due to resampling differences. In fact, we find that our model actually better mimics the content extraction and linearization of GPT-4o than its smaller counterpart GPT-4o mini.

When partitioning scores in low, medium, and high alignment buckets (Table 8), we find that most documents parsed with OLMOCR have medium to high alignment with GPT-4o. Increasing temperature unsurprisingly leads to a wider distribution of alignment scores, as noted by the increase of low matches for $\tau = 0.8$.

Model	Temperature τ	Alignment
GPT-4o (self-alignment)	*0.1*	*0.954*
GPT-4o mini	0.1	0.833
OLMOCR	0.8	0.859
OLMOCR	0.1	**0.875**

Table 7 Page-weighted alignment between GPT-4o, GPT-4o mini, and our fine-tuned model. We find that `olmOCR-7B-0225-preview` is more consistent with respect to its teacher than GPT-4o mini. Note that GPT-4o does not achieves a perfect alignment against itself due to the probabilistic nature of autoregressive decoding.

Name	Low match	Medium match	High match
GPT-4o (self alignment)	*38*	*218*	*965*
GPT-4o mini	214	478	529
olmOCR ($\tau = 0.1$)	158	363	700
olmOCR ($\tau = 0.8$)	195	390	636

Table 8 Match-up between olmOCR and different models compared to the `olmOCR-mix-0225` dataset. Low match indicates $< 70\%$ alignment, Medium match is 70-95% alignment, High match is $>95\%$ alignment.

C.2 Intrinsic Human Evaluation

Experimental setup To compare olmOCR against other common OCR methods, we collected pairwise human judgments of plain text produced by the three top ML-based PDF linearization tools—Marker, MinerU, and GOT-OCR 2.0—and calculating ELO ratings.

To create our evaluation set, we sample 2,017 new PDFs from the same distribution as used to create `olmOCR-mix-0225` and run each PDF through olmOCR and the linearization tools mentioned above. All other linearization tools were installed from either PyPI or Github according to their publicly available instructions as of January 14th, 2025. GOT-OCR 2.0 was configured in 'format' mode, but otherwise all comparisons were done against default settings.

We then sampled 2,000 comparison pairs (same PDF, different tool). We asked 11 data researchers and engineers at Ai2 to assess which output was the higher quality representation of the original PDF, focusing on reading order, comprehensiveness of content and representation of structured information. The user interface used is similar to that in Figure 6. Exact participant instructions are listed in Appendix C.3.

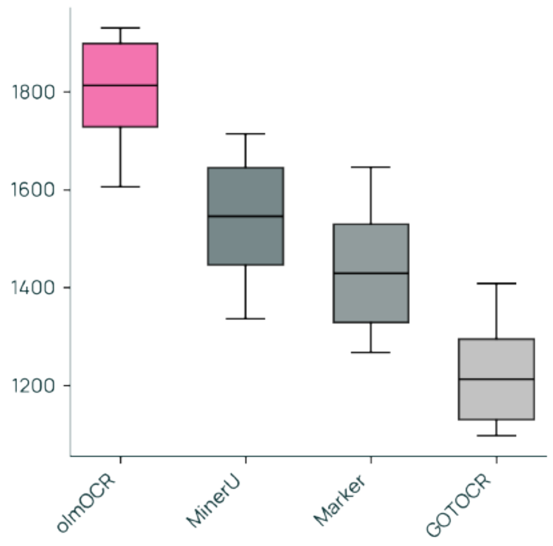


Figure 7 ELO ranking of olmOCR vs other popular PDF content extraction tools.

Evaluation results We collected a total of 452 judgments where a participant expressed a preference between two models (the remaining 1,548 pairs were either skipped for being too similar, or marked as invalid). On average, this is 75 judgments per pair of tools. We calculate ELO ratings starting from a base of 1500 and report the average of 100 simulations to avoid ordering effects in ELO calculations; for 95% confidence intervals, we use bootstrapping with 5000 resamples.

We visualize our results in Figure 7. olmOCR achieves an ELO score over 1800, far exceeding all other PDF linearization tools.

C.3 ELO Evaluation Instructions

In Section C.2, we asked participants to compare the output of various common OCR tools against olmOCR. Participants were given the instructions below, and presented with a document page, and the output of two random tools. They could then select which output was better, or select 'Both Good', 'Both Bad', or 'Invalid PDF' any of which would not count the comparison in the ELO ranking.

Instructions to participants

```
Compare the text in the two fields, and select which one better represents the contents of
    the document.

REMINDER: This is not about "the most faithful OCR", but "this OCR output seems really
    useful for training LMs"

- Does the text capture all of the meaningful content in the document in a natural order?
- Are the words correct (no weird incorrect words or split words)
- Is the whitepsace sensical?
- Is the useless header/footer content removed?
- Do the tables/equations look okay?

There is not a strict preference between Markdown and LaTeX, most importantly you should
    evaluate it on the text content, not which method was used to format it.

If you are not sure, or the document is in a language other than english, you can skip that
    entry, or mark "both good" "both bad", "invalid pdf".
```

ELO data

We compute pairwise win/loss statistics between models to estimate relative performance under head-to-head comparisons. As shown in Table 9, olmOCR consistently outperforms other models such as Marker, GOTOCR, and MinerU, with the highest win rate of 71.4% against MinerU.

Table 9 Pairwise Win/Loss Statistics Between Models

Model Pair	Wins	Win Rate (%)
olmOCR vs. Marker	49/31	**61.3**
olmOCR vs. GOTOCR	41/29	**58.6**
olmOCR vs. MinerU	55/22	**71.4**
Marker vs. MinerU	53/26	67.1
Marker vs. GOTOCR	45/26	63.4
GOTOCR vs. MinerU	38/37	50.7
Total	452	

D Deploying olmOCR

D.1 Inference Pipeline

To efficiently convert millions of documents, we develop the olmOCR pipeline using SGLang (Zheng et al., 2024) as the inference engine. The pipeline batches documents into work items of around 500 pages each. Each work item is then queued to run on a worker with access to a GPU for inference. Optionally, workers can coordinate using a shared cloud bucket[12], allowing for batch jobs that scale from single nodes to hundreds of nodes without the need for complicated queue management.

[12]We use Amazon Simple Storage Service (S3), but other cloud providers or network storage solutions could be easily used.

We summarize our efforts by comparing operational costs of olmOCR against other API and local models in Table 6. Overall, we find olmOCR to be significantly more efficient than other pipelines. It is over 32 times cheaper than GPT-4o in batch mode; compared to other purposed-built pipelines and models, olmOCR is over 6 times cheaper than MinerU, and $1/3^{rd}$ of the cost of marker.

To balance maintaining high GPU utilization while also ensuring work items are completed quickly, each worker queues up inference for all PDF pages in a work item simultaneously, and then waits until the SGLang server has no more pending requests before proceeding to another work item in the queue.

D.2 Increasing Robustness

We implement several heuristics to improve reliability of olmOCR without compromising its throughput.

Prompt format During inference, we use the same abbreviated prompt described in Section §2.3. This keeps the test time examples looking the same as what the model was trained on. If the additional tokens generated by document-anchoring cause the overall prompt to exceed 8,192 tokens, then we continue regenerating the document-anchoring tokens with exponentially lower character limits until the overall prompt is of acceptable length.

Retries Unlike when we created `olmOCR-mix-0225`, we do not enforce a specific JSON schema during inference on our fine-tuned model. This is for two reasons: first, we find that open source tools designed to force decode a sequence into a particular schema are unreliable, and that enforcing a schema which is even slightly off from what the model expects can cause generations to go out-of-domain or collapse into repetitions. Second, and most importantly, we note that, since the model was extensively fine-tuned on the structured output, it reliably adheres to the required schema without constraints. For the rare cases when JSON parsing fails, we simply retry generating from the same input sequence.

Rotations The output JSON schema includes fields for `is_rotation_valid` and `rotation_correction`. During inference, olmOCR pipeline reads these two fields and if `is_rotation_valid` is set to `true` it will rotate the page by the amount specified in `rotation_correction` and reprocess the page.

Decoding In developing olmOCR, the most common failure we experience is outputs degenerating into endless repetitions of the same token, line, or paragraph. This failure is caught automatically when the model's output either exceeds the maximum context length, or does not validate against our JSON schema. We find that increasing generation temperature from $\tau = 0.1$ up to $\tau = 0.8$ reduces the likelihood of repetitions occurring. Further, we modify olmOCR to reprocess failed pages up to N times, falling back to a plain text-based PDF extraction if the pipeline repeatedly fails. This last mitigation is aided by the fact that document-anchoring randomly samples which anchors to include in the prompt; thus, resampling can sometimes help the page process correctly by removing potentially problematic meta tokens.

We note that one one limitation of this approach is that, if retries occur often, the total generation throughput could be significantly reduced. Further, letting generations repeat up to maximum sequence length uses significant memory within SGLang. In future work, we plan to detect repeated generations sooner than at the maximum context length limit, and abort promptly.

E `olmOCR-mix-0225` and `olmOCR-7B-0225-preview` Prompts

E.1 `olmOCR-mix-0225` construction prompt for GPT-4o

The prompt below was used to create the silver dataset, which we refer to as `olmOCR-mix-0225` throughout the paper. This dataset consists of structured outputs generated by GPT-4o, using images of PDF pages along with additional layout-aware textual features produced by our document-anchoring pipeline. We use this synthetic data to fine-tune our model.

In this prompt, the placeholder `{base_text}` is replaced with the structured layout-aware text extracted from the PDF using document-anchoring. The prompt instructs GPT-4o to output the natural reading-order text of the page, while respecting document semantics, suppressing hallucinations, and formatting content like equations and tables appropriately.

```
Below is the image of one page of a PDF document, as well as some raw textual content that
    was previously extracted for it that includes position information for each image and
    block of text (The origin [0x0] of the coordinates is in the lower left corner of the
    image).
Just return the plain text representation of this document as if you were reading it
    naturally.
Turn equations into a LaTeX representation, and tables into markdown format. Remove the
    headers and footers, but keep references and footnotes.
Read any natural handwriting.
This is likely one page out of several in the document, so be sure to preserve any sentences
     that come from the previous page, or continue onto the next page, exactly as they are.
If there is no text at all that you think you should read, you can output null.
Do not hallucinate.
RAW_TEXT_START
{base_text}
RAW_TEXT_END
```

JSON Schema used to prompt GPT-4o

```
"json_schema": {
            "name": "page_response",
            "schema": {
                "type": "object",
                "properties": {
                    "primary_language": {
                        "type": ["string", "null"],
                        "description": "The primary language of the text using two-letter
                            codes or null if there is no text at all that you think you
                            should read.",
                    },
                    "is_rotation_valid": {
                        "type": "boolean",
                        "description": "Is this page oriented correctly for reading? Answer
                            only considering the textual content, do not factor in the
                            rotation of any charts, tables, drawings, or figures.",
                    },
                    "rotation_correction": {
                        "type": "integer",
                        "description": "Indicates the degree of clockwise rotation needed if
                             the page is not oriented correctly.",
                        "enum": [0, 90, 180, 270],
                        "default": 0,
                    },
                    "is_table": {
                        "type": "boolean",
                        "description": "Indicates if the majority of the page content is in
                            tabular format.",
                    },
                    "is_diagram": {
                        "type": "boolean",
                        "description": "Indicates if the majority of the page content is a
                            visual diagram.",
                    },
                    "natural_text": {
                        "type": ["string", "null"],
                        "description": "The natural text content extracted from the page.",
                    },
                },
                "additionalProperties": False,
                "required": [
                    "primary_language",
                    "is_rotation_valid",
                    "rotation_correction",
                    "is_table",
                    "is_diagram",
                    "natural_text",
                ],
            },
            "strict": True,
        },
```

E.2 `olmOCR-7B-0225-preview` prompt

The prompt below is used to draw responses from our fine-tuned model during inference. As before, the placeholder `{base_text}` is replaced with the output of the DOCUMENT-ANCHORING pipeline i.e., layout-aware textual features extracted from the PDF page.

```
Below is the image of one page of a document, as well as some raw textual content that was
    previously extracted for it.
Just return the plain text representation of this document as if you were reading it
    naturally.
Do not hallucinate.
RAW_TEXT_START
{base_text}
RAW_TEXT_END
```

E.3 `olmOCR-mix-0225` Classification Prompt

The prompt and structured schema below was used to classify a sample of documents from `olmOCR-mix-0225` as reported in Table 2.

```
This is an image of a document page, please classify it into one of the following categories
     that best overall summarizes its nature: academic, legal, brochure, slideshow, table,
    diagram, or other. Also determine the primary language of the document and your
    confidence in the classification (0-1).
```

```
class DocumentCategory(str, Enum):
    ACADEMIC = "academic"
    LEGAL = "legal"
    BROCHURE = "brochure"
    SLIDESHOW = "slideshow"
    TABLE = "table"
    DIAGRAM = "diagram"
    OTHER = "other"

class DocumentClassification(BaseModel):
    category: DocumentCategory
    language: str
    confidence: float
```

E.4 `olmOCR-mix-0225` PII Prompt

We implemented comprehensive prompting for detecting personally identifiable information (PII) in the documents while cleaning the `olmOCR-mix-0225`:

```
You are a document analyzer that identifies Personally Identifiable Information
(PII) in documents.
Your task is to analyze the provided document image and determine:
1. Whether the document is intended for public release or dissemination
   (e.g., research paper, public report, etc.)
2. If the document contains any PII

For PII identification, follow these specific guidelines:
IDENTIFIERS FOR PII:
The following are considered identifiers that can make information PII:
- Names (full names, first names, last names, nicknames)
- Email addresses
- Phone numbers
```

```
PII THAT MUST CO-OCCUR WITH AN IDENTIFIER:
The following types of information should ONLY be marked as PII if they occur
ALONGSIDE an identifier (commonly, a person's name):
- Addresses (street address, postal code, etc.)
- Biographical Information (date of birth, place of birth, gender, sexual
  orientation, race, ethnicity, citizenship/immigration status, religion)
- Location Information (geolocations, specific coordinates)
- Employment Information (job titles, workplace names, employment history)
- Education Information (school names, degrees, transcripts)
- Medical Information (health records, diagnoses, genetic or neural data)

PII THAT OCCURS EVEN WITHOUT AN IDENTIFIER:
The following should ALWAYS be marked as PII even if they do not occur
alongside an identifier:
- Government IDs (Social Security Numbers, passport numbers, driver's license
  numbers, tax IDs)
- Financial Information (credit card numbers, bank account/routing numbers)
- Biometric Data (fingerprints, retina scans, facial recognition data,
  voice signatures)
- Login information (ONLY mark as PII when a username, password, and login
  location are present together)

If the document is a form, then only consider fields which are filled out
with specific values as potential PII.
If this page does not itself contain PII, but references documents
(such as curriculum vitae, personal statements) that typically contain PII,
then do not mark it as PII.
Only consider actual occurrences of the PII within the document shown.
```

F Further details of olmOCR-Bench

F.1 Data sources

See Table 10 for further details about PDFs selected for each olmOCR-Bench category, the source, and the processing.

Table 10 Document source category breakdown of olmOCR-Bench

Category	PDFs	Tests	Source	Extraction Method
arXiv_math	522	2,927	arXiv	Dynamic programming alignment
old_scans_math	36	458	Internet Archive	Script-generated + manual rules
tables_tests	188	1,020	Internal repository	`gemini-flash-2.0`
old_scans	98	526	Library of Congress	Manual rules
headers_footers	266	753	Internal repository	DocLayout-YOLO + `gemini-flash-2.0`
multi_column	231	884	Internal repository	`claude-sonnet-3.7` + HTML rendering
long_tiny_text	62	442	Internet Archive	`gemini-flash-2.0`
Total	1,403	7,010	Multiple sources	

F.2 Prompting Strategies and Implementation Details

This section provides comprehensive documentation of the prompting techniques and design strategies to make olmOCR-Bench. These prompting approaches were critical in generating test cases while utilizing LLMs and ensuring consistency across document categories.

F.2.1 Mathematical Expressions

For generating mathematical expression test cases from old scans, we employed direct prompts focused on precision. This concise prompt architecture proved effective in extracting LaTeX representations minimizing hallucination. The explicit instruction to use standard LaTeX delimiters ($$) ensured consistent formatting across the olmOCR-Bench.

```
Please extract the mathematical equations from the document without
omission. Always output the mathematical equations as Latex escaped
with $$. Do not hallucinate.
```

F.2.2 Multi-column

For Multi-column documents, we utilized a two-stage prompting strategy. The initial analytical stage established structural context:

```
Analyze this document and provide a detailed assessment of its structure.
Focus on the layout, headings, footers, and any complex formatting.
Please be precise.
```

This preliminary analysis was incorporated into a subsequent HTML rendering prompt:

```
Render this document as clean, semantic HTML. Here is the analysis of the
document structure:

{analysis_text}

Requirements:
1. Use appropriate HTML tags for headings, paragraphs, and lists.
2. Use <header> and <footer> for top and bottom content.
3. For images, use a placeholder <div> with class 'image'.
4. Render math equations inline using \( \) or \[ \].
```

```
5. Preserve any multi-column layout using CSS flexbox or grid.
6. The viewport is fixed at {png_width // 2}x{png_height // 2} pixels.

Enclose your HTML in a html code block.
```

This approach significantly helped in layout preservation in complex documents by providing explicit dimensional constraints and structural information.

F.2.3 PII Detection and Filtering

We use the same PII detection and filtering as for construction `olmOCR-mix-0225`; see Appendix E.4.

F.2.4 Cleaning Mathematical Expressions

Mathematical expression verification employed specialized prompting for validating equation presence and accuracy:

```
This is a mathematical expression verification task.
I'm showing you a page from a PDF document containing mathematical expressions.
Please verify if the following LaTeX expression:
{latex_expression}
appears correctly in the document.
Respond with a JSON object containing:
1. "status": "correct" or "incorrect"
2. "confidence": a value between 0 and 1 representing your confidence in the answer
3. "explanation": a brief explanation of why you believe the expression is correct or
    incorrect
Focus specifically on checking if this exact mathematical expression appears in the document
    .
```

F.2.5 Cleaning Reading Order Tests

For natural reading order test cases, we implemented below verification prompt to ensure appropriate text segment relationships:

```
Does the text in the 'before' field and the 'after' field appear in the same region of the
    page?
Look at the PDF image and determine if these texts are located near each other or in
    completely
different parts of the page. Different regions could be the captions for different images,
    or
inside of different insets or tables. However, appearing the same column of text, or in the
naturally flowing next column of text is close enough.

Before: {before_text}

After: {after_text}

Respond with 'YES' if they appear in the same region or column, and 'NO' if they appear in
different regions. Then explain your reasoning in 1-2 sentences.
```

F.2.6 Header and Footer Verification

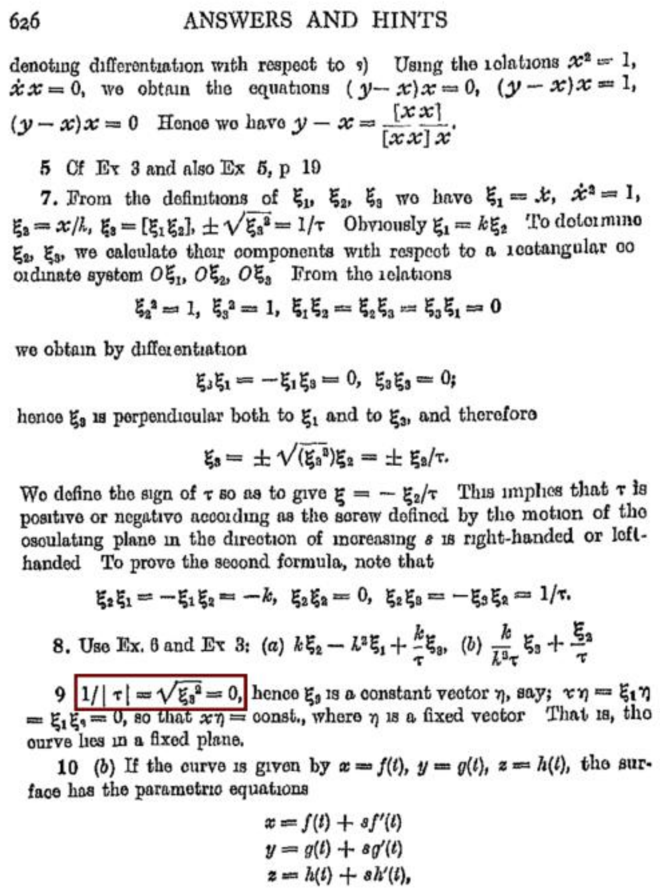
For validating header and footer text identification, we employed JSON-structured verification prompts:

```
This is a header and footer verification task.
I'm showing you a page from a PDF document containing headers and footers text.
Please verify if the headers or footers is exactly matches the below text.
{header_footer_text}
Respond with a JSON object containing:
1. "status": "correct" or "incorrect"
2. "confidence": a value between 0 and 1 representing your confidence in the answer
3. "explanation": a brief explanation of why you believe the text is correct or incorrect
Focus specifically on checking if this exact header or footer expression appears in the
    document.
```

Our prompting strategy deliberately requested different output formats for different content types (Markdown for general text, LaTeX for equations, HTML for tables) to optimize representation fidelity across diverse document elements. Low temperature settings (typically 0.1) was maintained across all the prompt executions to ensure reproducible outputs, particularly important for establishing consistent test cases.

F.3 Sample Test Classes

Below are are few examples taken from olmOCR-Bench



626 ANSWERS AND HINTS

denoting differentiation with respect to s) Using the relations $\mathfrak{x}^2 = 1$, $\dot{\mathfrak{x}}\mathfrak{x} = 0$, we obtain the equations $(\mathfrak{y} - \mathfrak{x})\mathfrak{x} = 0$, $(\mathfrak{y} - \mathfrak{x})\dot{\mathfrak{x}} = 1$, $(\mathfrak{y} - \mathfrak{x})\ddot{\mathfrak{x}} = 0$ Hence we have $\mathfrak{y} - \mathfrak{x} = \frac{[\dot{\mathfrak{x}}\ddot{\mathfrak{x}}]}{[\dot{\mathfrak{x}}\ddot{\mathfrak{x}}]\dot{\mathfrak{x}}}$.

5 Cf Ex 3 and also Ex 5, p 19

7. From the definitions of ξ_1, ξ_2, ξ_3 we have $\xi_1 = \dot{\mathfrak{x}}$, $\dot{\mathfrak{x}}^2 = 1$, $\xi_2 = \ddot{\mathfrak{x}}/k$, $\xi_3 = [\xi_1 \xi_2]$, $\pm\sqrt{\dot{\xi}_3^2} = 1/\tau$ Obviously $\dot{\xi}_1 = k\xi_2$ To determine $\dot{\xi}_2$, $\dot{\xi}_3$, we calculate their components with respect to a rectangular co ordinate system $O\xi_1$, $O\xi_2$, $O\xi_3$ From the relations

$$\xi_2^2 = 1,\ \xi_3^2 = 1,\ \xi_1\xi_2 = \xi_2\xi_3 = \xi_3\xi_1 = 0$$

we obtain by differentiation

$$\dot{\xi}_3\xi_1 = -\dot{\xi}_1\xi_3 = 0,\ \dot{\xi}_3\xi_3 = 0;$$

hence $\dot{\xi}_3$ is perpendicular both to ξ_1 and to ξ_3, and therefore

$$\dot{\xi}_3 = \pm\sqrt{(\dot{\xi}_3^2)}\xi_2 = \pm\,\xi_2/\tau.$$

We define the sign of τ so as to give $\dot{\xi} = -\,\xi_2/\tau$ This implies that τ is positive or negative according as the screw defined by the motion of the osculating plane in the direction of increasing s is right-handed or left-handed To prove the second formula, note that

$$\dot{\xi}_2\xi_1 = -\xi_1\dot{\xi}_2 = -k,\ \dot{\xi}_2\xi_2 = 0,\ \dot{\xi}_2\xi_3 = -\dot{\xi}_3\xi_2 = 1/\tau.$$

8. Use Ex. 6 and Ex 3: (a) $\dot{k}\xi_2 - k^2\xi_1 + \frac{k}{\tau}\xi_3$, (b) $\frac{\dot{k}}{k^2\tau}\,\xi_3 + \frac{\xi_2}{\tau}$

9 $\boxed{1/|\tau| = \sqrt{\dot{\xi}_3^2} = 0,}$ hence ξ_3 is a constant vector η, say; $\dot{\mathfrak{x}}\eta = \xi_1\eta = \xi_1\xi_3 = 0$, so that $\mathfrak{x}\eta =$ const., where η is a fixed vector That is, the curve lies in a fixed plane.

10 (b) If the curve is given by $x = f(t)$, $y = g(t)$, $z = h(t)$, the surface has the parametric equations

$$x = f(t) + sf'(t)$$
$$y = g(t) + sg'(t)$$
$$z = h(t) + sh'(t),$$

Figure 8 Sample visualization from old_scans_math. The OCR output for the highlighted equation should be: `1/|\tau| = \sqrt{\xi_{3}^{2}} = 0`

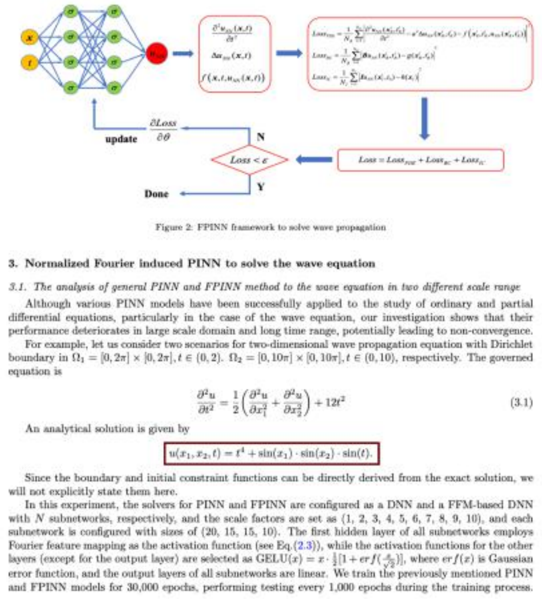

update
$\frac{\partial Loss}{\partial \theta}$
N
$Loss < \varepsilon$
Y
Done
$Loss = Loss_{PDE} + Loss_{BC} + Loss_{IC}$

Figure 2: FPINN framework to solve wave propagation

3. Normalized Fourier induced PINN to solve the wave equation

3.1. The analysis of general PINN and FPINN method to the wave equation in two different scale range

Although various PINN models have been successfully applied to the study of ordinary and partial differential equations, particularly in the case of the wave equation, our investigation shows that their performance deteriorates in large scale domain and long time range, potentially leading to non-convergence.

For example, let us consider two scenarios for two-dimensional wave propagation equation with Dirichlet boundary in $\Omega_1 = [0, 2\pi] \times [0, 2\pi], t \in (0, 2)$. $\Omega_2 = [0, 10\pi] \times [0, 10\pi], t \in (0, 10)$, respectively. The governed equation is

$$\frac{\partial^2 \mathrm{u}}{\partial t^2} = \frac{1}{2}\left(\frac{\partial^2 \mathrm{u}}{\partial x_1^2} + \frac{\partial^2 \mathrm{u}}{\partial x_2^2}\right) + 12t^2 \tag{3.1}$$

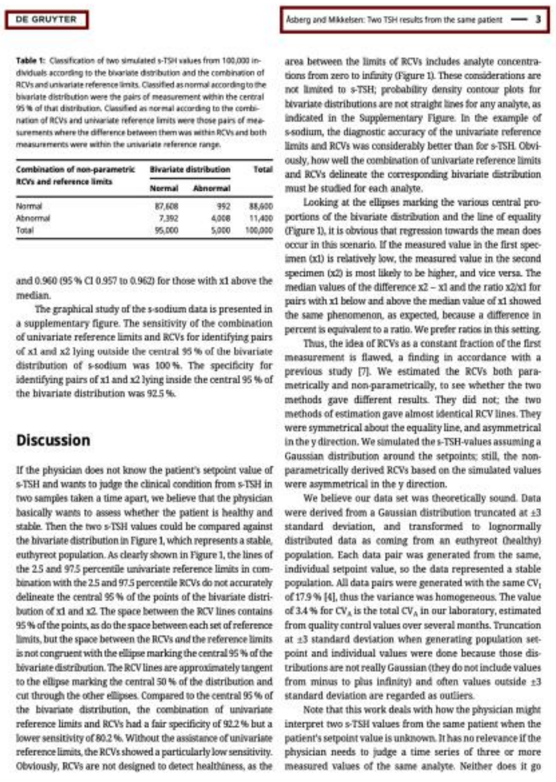
An analytical solution is given by

$$\boxed{\mathrm{u}(x_1, x_2, t) = t^4 + \sin(x_1) \cdot \sin(x_2) \cdot \sin(t).}$$

Since the boundary and initial constraint functions can be directly derived from the exact solution, we will not explicitly state them here.

In this experiment, the solvers for PINN and FPINN are configured as a DNN and a FFM-based DNN with N subnetworks, respectively, and the scale factors are set as (1, 2, 3, 4, 5, 6, 7, 8, 9, 10), and each subnetwork is configured with sizes of (20, 15, 15, 10). The first hidden layer of all subnetworks employs Fourier feature mapping as the activation function (see Eq.(2.3)), while the activation functions for the other layers (except for the output layer) are selected as $\mathrm{GELU}(x) = x \cdot \frac{1}{2}[1 + erf(\frac{x}{\sqrt{2}})]$, where $erf(x)$ is Gaussian error function, and the output layers of all subnetworks are linear. We train the previously mentioned PINN and FPINN models for 30,000 epochs, performing testing every 1,000 epochs during the training process.

Figure 9 Sample visualization of a math equation from arXiv_math. The OCR output for the highlighted equation should be: `u(x_{1},x_{2},t)=t^{4} + \text{sin}(x_{1}) \cdot \text{sin}(x_{2}) \cdot \text{sin}(t)`



DE GRUYTER

Asberg and Mikkelsen: Two TSH results from the same patient — 3

Table 1: Classification of two simulated s-TSH values from 100,000 individuals according to the bivariate distribution and the combination of RCVs and univariate reference limits. Classified as normal according to the bivariate distribution were the pairs of measurement within the central 95 % of that distribution. Classified as normal according to the combination of RCVs and univariate reference limits were those pairs of measurements where the difference between them was within RCVs and both measurements were within the univariate reference range.

Combination of non-parametric RCVs and reference limits	Bivariate distribution		Total
	Normal	Abnormal	
Normal	87,608	992	88,600
Abnormal	7,392	4,008	11,400
Total	95,000	5,000	100,000

and 0.960 (95 % CI 0.957 to 0.962) for those with x1 above the median.

The graphical study of the s-sodium data is presented in a supplementary figure. The sensitivity of the combination of univariate reference limits and RCVs for identifying pairs of x1 and x2 lying outside the central 95 % of the bivariate distribution of s-sodium was 100 %. The specificity for identifying pairs of x1 and x2 lying inside the central 95 % of the bivariate distribution was 92.5 %.

Discussion

If the physician does not know the patient's setpoint value of s-TSH and wants to judge the clinical condition from s-TSH in two samples taken a time apart, we believe that the physician basically wants to assess whether the patient is healthy and stable. Then the two s-TSH values could be compared against the bivariate distribution in Figure 1, which represents a stable, euthyreot population. As clearly shown in Figure 1, the lines of the 2.5 and 97.5 percentile univariate reference limits in combination with the 2.5 and 97.5 percentile RCVs do not accurately delineate the central 95 % of the points of the bivariate distribution of x1 and x2. The space between the RCV lines contains 95 % of the points, as do the space between each set of reference limits, but the space between the RCVs and the reference limits is not congruent with the ellipse marking the central 95 % of the bivariate distribution. The RCV lines are approximately tangent to the ellipse marking the central 50 % of the distribution and cut through the other ellipses. Compared to the central 95 % of the bivariate distribution, the combination of univariate reference limits and RCVs had a fair specificity of 92.2 % but a lower sensitivity of 80.2 %. Without the assistance of univariate reference limits, the RCVs showed a particularly low sensitivity. Obviously, RCVs are not designed to detect healthiness, as the area between the limits of RCVs includes analyte concentrations from zero to infinity (Figure 1). These considerations are not limited to s-TSH; probability density contour plots for bivariate distributions are not straight lines for any analyte, as indicated in the Supplementary Figure. In the example of s-sodium, the diagnostic accuracy of the univariate reference limits and RCVs was considerably better than for s-TSH. Obviously, how well the combination of univariate reference limits and RCVs delineate the corresponding bivariate distribution must be studied for each analyte.

Looking at the ellipses marking the various central proportions of the bivariate distribution and the line of equality (Figure 1), it is obvious that regression towards the mean does occur in this scenario. If the measured value in the first specimen (x1) is relatively low, the measured value in the second specimen (x2) is most likely to be higher, and vice versa. The median values of the difference x2 – x1 and the ratio x2/x1 for pairs with x1 below and above the median value of x1 showed the same phenomenon, as expected, because a difference in percent is equivalent to a ratio. We prefer ratios in this setting.

Thus, the idea of RCVs as a constant fraction of the first measurement is flawed, a finding in accordance with a previous study [7]. We estimated the RCVs both parametrically and non-parametrically, to see whether the two methods gave different results. They did not; the two methods of estimation gave almost identical RCV lines. They were symmetrical about the equality line, and asymmetrical in the y direction. We simulated the s-TSH-values assuming a Gaussian distribution around the setpoints; still, the non-parametrically derived RCVs based on the simulated values were asymmetrical in the y direction.

We believe our data set was theoretically sound. Data were derived from a Gaussian distribution truncated at ±3 standard deviation, and transformed to lognormally distributed data as coming from an euthyreot (healthy) population. Each data pair was generated from the same, individual setpoint value, so the data represented a stable population. All data pairs were generated with the same CV_I of 17.9 % [4], thus the variance was homogeneous. The value of 3.4 % for CV_A is the total CV_A in our laboratory, estimated from quality control values over several months. Truncation at ±3 standard deviation when generating population setpoint and individual values were done because those distributions are not really Gaussian (they do not include values from minus to plus infinity) and often values outside ±3 standard deviation are regarded as outliers.

Note that this work deals with how the physician might interpret two s-TSH values from the same patient when the patient's setpoint value is unknown. It has no relevance if the physician needs to judge a time series of three or more measured values of the same analyte. Neither does it go

Figure 10 Sample visualization of `headers_footers`. We want the OCR to skip the document headers and page number.

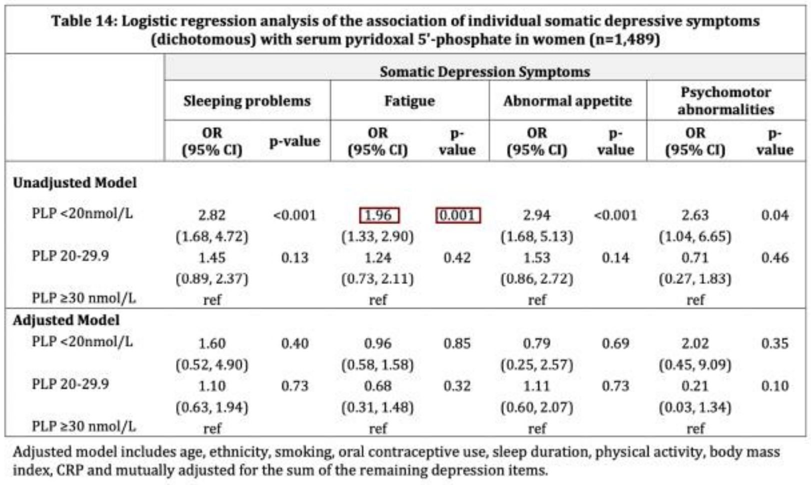
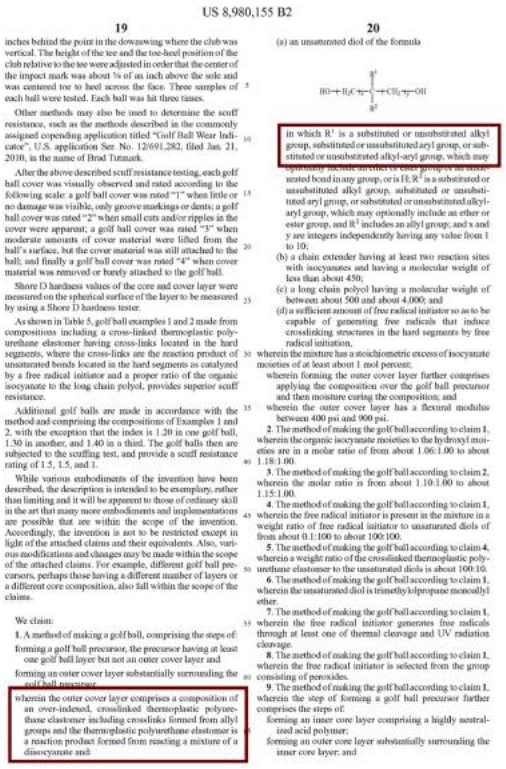


Table 14: Logistic regression analysis of the association of individual somatic depressive symptoms (dichotomous) with serum pyridoxal 5'-phosphate in women (n=1,489)

	Somatic Depression Symptoms							
	Sleeping problems		Fatigue		Abnormal appetite		Psychomotor abnormalities	
	OR (95% CI)	p-value	OR (95% CI)	p-value	OR (95% CI)	p-value	OR (95% CI)	p-value
Unadjusted Model								
PLP <20nmol/L	2.82 (1.68, 4.72)	<0.001	1.96 (1.33, 2.90)	0.001	2.94 (1.68, 5.13)	<0.001	2.63 (1.04, 6.65)	0.04
PLP 20-29.9	1.45 (0.89, 2.37)	0.13	1.24 (0.73, 2.11)	0.42	1.53 (0.86, 2.72)	0.14	0.71 (0.27, 1.83)	0.46
PLP ≥30 nmol/L	ref		ref		ref		ref	
Adjusted Model								
PLP <20nmol/L	1.60 (0.52, 4.90)	0.40	0.96 (0.58, 1.58)	0.85	0.79 (0.25, 2.57)	0.69	2.02 (0.45, 9.09)	0.35
PLP 20-29.9	1.10 (0.63, 1.94)	0.73	0.68 (0.31, 1.48)	0.32	1.11 (0.60, 2.07)	0.73	0.21 (0.03, 1.34)	0.10
PLP ≥30 nmol/L	ref		ref		ref		ref	

Adjusted model includes age, ethnicity, smoking, oral contraceptive use, sleep duration, physical activity, body mass index, CRP and mutually adjusted for the sum of the remaining depression items.

Figure 11 Sample visualization of `table_tests`. We want the OCR to predict that cell 1.96 is to the left of cell 0.001.



US 8,980,155 B2

19

inches behind the point in the downswing where the club was vertical. The height of the tee and the toe-heel position of the club relative to the tee were adjusted in order that the center of the impact mark was about ⅛ of an inch above the sole and was centered toe to heel across the face. Three samples of each ball were tested. Each ball was hit three times.

Other methods may also be used to determine the scuff resistance, such as the methods described in the commonly assigned copending application titled "Golf Ball Wear Indicator", U.S. application Ser. No. 12/691,282, filed Jan. 21, 2010, in the name of Brad Tutmark.

After the above described scuff resistance testing, each golf ball cover was visually observed and rated according to the following scale: a golf ball cover was rated "1" when little or no damage was visible, only groove markings or dents; a golf ball cover was rated "2" when small cuts and/or ripples in the cover were apparent; a golf ball cover was rated "3" when moderate amounts of cover material were lifted from the ball's surface, but the cover material was still attached to the ball; and finally a golf ball cover was rated "4" when cover material was removed or barely attached to the golf ball.

Shore D hardness values of the core and cover layer were measured on the spherical surface of the layer to be measured by using a Shore D hardness tester.

As shown in Table 5, golf ball examples 1 and 2 made from compositions including a cross-linked thermoplastic polyurethane elastomer having cross-links located in the hard segments, where the cross-links are the reaction product of unsaturated bonds located in the hard segments as catalyzed by a free radical initiator and a proper ratio of the organic isocyanate to the long chain polyol, provides superior scuff resistance.

Additional golf balls are made in accordance with the method and comprising the compositions of Examples 1 and 2, with the exception that the index is 1.20 in one golf ball, 1.30 in another, and 1.40 in a third. The golf balls then are subjected to the scuffing test, and provide a scuff resistance rating of 1.5, 1.5, and 1.

While various embodiments of the invention have been described, the description is intended to be exemplary, rather than limiting and it will be apparent to those of ordinary skill in the art that many more embodiments and implementations are possible that are within the scope of the invention. Accordingly, the invention is not to be restricted except in light of the attached claims and their equivalents. Also, various modifications and changes may be made within the scope of the attached claims. For example, different golf ball precursors, perhaps those having a different number of layers or a different core composition, also fall within the scope of the claims.

We claim:

1. A method of making a golf ball, comprising the steps of:

forming a golf ball precursor, the precursor having at least one golf ball layer but not an outer cover layer and

forming an outer cover layer substantially surrounding the golf ball precursor

wherein the outer cover layer comprises a composition of an over-indexed, crosslinked thermoplastic polyurethane elastomer including crosslinks formed from allyl groups and the thermoplastic polyurethane elastomer is a reaction product formed from reacting a mixture of a diisocyanate and:

20

(a) an unsaturated diol of the formula

$$HO\!-\!\!\left(H_2C\right)_x\!-\!\underset{R^2}{\overset{R^1}{C}}\!-\!\left(CH_2\right)_y\!-\!OH$$

in which R^1 is a substituted or unsubstituted alkyl group, substituted or unsubstituted aryl group, or substituted or unsubstituted alkyl-aryl group, which may optionally include an ether or ester group or an unsaturated bond in any group, or is H; R^2 is a substituted or unsubstituted alkyl group, substituted or unsubstituted aryl group, or substituted or unsubstituted alkyl-aryl group, which may optionally include an ether or ester group, and R^2 includes an allyl group; and x and y are integers independently having any value from 1 to 10;

(b) a chain extender having at least two reaction sites with isocyanates and having a molecular weight of less than about 450;

(c) a long chain polyol having a molecular weight of between about 500 and about 4,000; and

(d) a sufficient amount of free radical initiator so as to be capable of generating free radicals that induce crosslinking structures in the hard segments by free radical initiation,

wherein the mixture has a stoichiometric excess of isocyanate moieties of at least about 1 mol percent;

wherein forming the outer cover layer further comprises applying the composition over the golf ball precursor and then moisture curing the composition; and

wherein the outer cover layer has a flexural modulus between 400 psi and 900 psi.

2. The method of making the golf ball according to claim 1, wherein the organic isocyanate moieties to the hydroxyl moieties are in a molar ratio of from about 1.06:1.00 to about 1.18:1.00.

3. The method of making the golf ball according to claim 2, wherein the molar ratio is from about 1.10:1.00 to about 1.15:1.00.

4. The method of making the golf ball according to claim 1, wherein the free radical initiator is present in the mixture in a weight ratio of free radical initiator to unsaturated diols of from about 0.1:100 to about 100:100.

5. The method of making the golf ball according to claim 4, wherein a weight ratio of the crosslinked thermoplastic polyurethane elastomer to the unsaturated diols is about 100:10.

6. The method of making the golf ball according to claim 1, wherein the unsaturated diol is trimethylolpropane monoallyl ether.

7. The method of making the golf ball according to claim 1, wherein the free radical initiator generates free radicals through at least one of thermal cleavage and UV radiation cleavage.

8. The method of making the golf ball according to claim 1, wherein the free radical initiator is selected from the group consisting of peroxides.

9. The method of making the golf ball according to claim 1, wherein the step of forming a golf ball precursor further comprises the steps of:

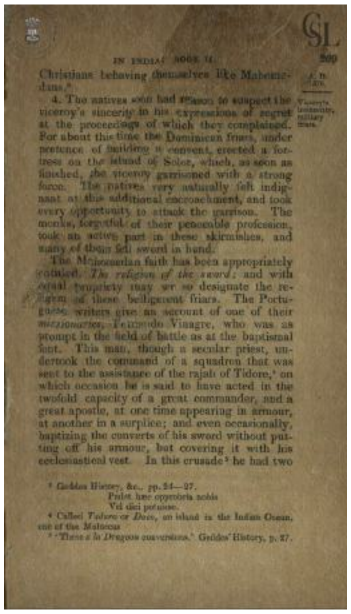
forming an inner core layer comprising a highly neutralized acid polymer;

forming an outer core layer substantially surrounding the inner core layer; and

Figure 12 Sample visualization of `reading_order`. The reading order should start with the left column before moving to the right column.

G Example output

Below are some sample outputs on particularly challenging data. olmOCR, MinerU, GOT-OCR 2.0 and Marker run with default settings.



IN INDIA: BOOK II. 269

Christians behaving themselves like Mahomedans.[3]

A.D. 1670.

Viceroy's insincerity, military friars.

4. The natives soon had reason to suspect the viceroy's sincerity in his expressions of regret at the proceedings of which they complained. For about this time the Dominican friars, under pretence of building a convent, erected a fortress on the island of Solor, which, as soon as finished, the viceroy garrisoned with a strong force. The natives very naturally felt indignant at this additional encroachment, and took every opportunity to attack the garrison. The monks, forgetful of their peaceable profession, took an active part in these skirmishes, and many of them fell sword in hand.

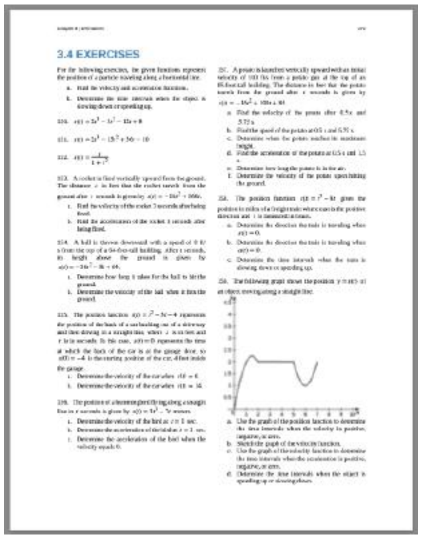
The Mahomedan faith has been appropriately entitled, *The religion of the sword*; and with equal propriety may we so designate the religion of these belligerent friars. The Portuguese writers give an account of one of their *missionaries*, Fernando Vinagre, who was as prompt in the field of battle as at the baptismal font. This man, though a secular priest, undertook the command of a squadron that was sent to the assistance of the rajah of Tidore,[4] on which occasion he is said to have acted in the twofold capacity of a great commander, and a great apostle, at one time appearing in armour, at another in a surplice; and even occasionally, baptizing the converts of his sword without putting off his armour, but covering it with his ecclesiastical vest. In this crusade[5] he had two

[3] Geddes History, &c., pp. 24—27.
Pudet hæc opprobria nobis
Vel dici potuisse.

[4] Called *Tadura* or *Daco*, an island in the Indian Ocean, one of the Moluccas.

[5] "These *à la Dragoon* conversions." Geddes' History, p. 27.

olmOCR	MinerU	GOT-OCR 2.0	Marker
Christians behaving themselves like Mahomedans. 4. The natives soon had reason to suspect the viceroy's sincerity in his expressions of regret at the proceedings of which they complained. For about this time the Dominican friars, under pretence of building a convent, erected a fortress on the island of Solor, which, as soon as finished, the viceroy garrisoned with a strong force. The natives very naturally felt indignant at this additional encroachment, and took every opportunity to attack the garrison. The monks, forgetful of their peaceable profession, took an active part in these skirmishes, and many of them fell sword in hand. The Mahomedan faith has been appropriately entitled, The religion of the sword; and with equal propriety may we so designate the religion of these belligerent friars. The Portuguese writers give an account of one of their missionaries, Fernando Vinagre, who was as prompt in the field of battle as at the baptismal font. This man, though a secular priest, undertook the command of a squadron that was sent to the assistance of the rajah of Tidore, on which occasion he is said to have acted in the twofold capacity of a great commander, and a great apostle, at one time appearing in armour, at another in a surplice; and even occasionally, baptizing the converts of his sword without putting off his armour, but covering it with his ecclesiastical vest. In this crusade he had two	ININDIASY BOOKU Christians bchaving.themselves like Mahome dans.3 4.The natives soon had reason to suspect ihe viceroy's sincerity in his expressions of regret at the proceedings of which they complained. For about this time the Dominican friars,under pretenceof building a convent,erected a for tress on the island of Solorwhich,as soon as finishedthe viceroy garrisoned with a strong force. The natives very naturally felt indig nant at this additional encroachment, and took every pportunity to attack the garrison.The monks,forgetful of their peaceable profession took an activa part in these skirmishes, and many of tbein feil sword in hand. TheMahornedan faithhas been appropriately ntitled.The religion of the swordand with equal propriety may we so designate the region of these belligerent friars.The Portugueswriters give an account of one of their missionarzes,femando Vinagre,who was as prompt in the field of battle as at the baptismal font. This man, though a secular priest, undertook the command of a squadron that was sent to the assistance of the rajah of Tidore,4 on which occasion he is said to have acted in the twofold capacity of a great commander, and a great apostle, at one time appearing in armour, at another in a surplice;and even occasionally baptizing the converts of his sword without put ting off his armour, but covering it with his ecclesiastical vest.In this crusadehe had two	IN INDIA: BOOK U 269 Christians behaving themselves like Mahome- 1670. 4. The natives son had reason to suspect the Viceroy' s vice roy' s sincerity in his expressions of regret in s in e eri ty at the proceedings of which they complained. fl it ars. For about this time the Dominican f mars, under pre ten ce of building a convent, erected a for- tress on the island of Sol or, which, as soon as finished, the vice roy garrisoned with a strong force. The natives very naturally felt indig- nant at this additional encroachment, and took every opportunity to attack the garrison. The monks, forgetful of their peaceable profession, took an active part in these skirmishes, and many of the n fell sword in hand. The Mh on med an faith has been appropriately entitled. The religion of the sword; and with e ral Tropriety may we so designate the re- gian of these belligerent friars. The Port u- gue s writers give an account of one of their mission are s, Fer endo Vina gre, who was as prompt in the fe ld of battle as at the baptismal font. This man, though a secular priest, un- der took the command of a squadron that was sent to the assistance of the rajah of Tidore, on which occasion he is said to have acted in the twofold capacity of a great commander, and a great apostle, at one time appearing in armour, at another in a surplice; and even occasionally, baptizing the converts of his sword without put- ting off his armour, but covering it with his ecclesiastical vest. In this crusade he had two 3 Ged des History, & c. , pp. 24-27. P ude th aec opp rob ria nobis Vel die ipo tui sse. Called Tadur u or Daco, an island in the Indian Ocean, one of the Mol ucc as These a laDra goon conversions. Ged des History, p. 27.	## **IN INDIA *** BOOK TI. S69 Christians behaving themselves like Ma borne- a. dans.3 ."5/0- *t>.* The natives soon had reason to suspect the viceroy, viceroy's sincerity in his expressions of regret at the proceedings of which they complained. "n."' For about this time the Dominican friars, under pretence of building a. convent, erected a fortress on the island of Sol or, which, as soon as finished, the viceroy garrisoned with a strong force. The natives' very naturally felt indig-S nant at this additional encroachment, and took every opportunity to attack the garrison. The monks, forgetful/ of their peaceable profession, took an active part in these skirmishes, and many of tbg.tr fell sword in hand. The i'lfinomedan faith has been appropriately entitled., 'The religion of the sword',; and with equal propriety may we so designate the re- . i'gv.m of these belligerent friars. The Portugu writers give an account of one of their 'missionaries,' Fernando Vinagre, who was as prompt in the field of battle as at the baptismal font. This man, though a secular priest, undertook the command of a squadron that was I sent to the assistance of the rajah of Tidore,4 on which occasion he is said to have acted in the twofold capacity of a great commander, and a great apostle, at one time appearing in armour, ; at another in a surplice; and even occasionally, baptizing the converts of his sword without putting off his armour, but covering it with his ecclesiastical vest. In this crusade5 he had two 3 Geddes History, &c., pp. 24—27. Pudet hæc opprobria nobis Vel dici potuisse. 4 Called 'T a d u ra' or 'D a c o,' an island in the Indian Ocean, one of the Moluccas 5 'These 'a la D ra g o o n' conversions.' Geddes' History, p. 27.



olmOCR

3.4 EXERCISES
For the following exercises, the given functions represent the position of a particle traveling along a horizontal line.
a. Find the velocity and acceleration functions.
b. Determine the time intervals when the object is slowing down or speeding up.
150. $s(t) = 2t^3 - 3t^2 - 12t + 8$
151. $s(t) = 2t^3 - 15t^2 + 36t - 10$
152. $s(t) = \frac{t}{1+t^2}$
153. A rocket is fired vertically upward from the ground. The distance s in feet that the rocket travels from the ground after t seconds is given by $s(t) = -16t^2 + 560t$.
a. Find the velocity of the rocket 3 seconds after being fired.
b. Find the acceleration of the rocket 3 seconds after being fired.
154. A ball is thrown downward with a speed of 8 ft/s from the top of a 64-foot-tall building. After t seconds, its height above the ground is given by $s(t) = -16t^2 - 8t + 64$.
a. Determine how long it takes for the ball to hit the ground.
b. Determine the velocity of the ball when it hits the ground.
155. The position function $s(t) = t^2 - 3t - 4$ represents the position of the back of a car backing out of a driveway and then driving in a straight line, where s is in feet and t is in seconds. In this case, $s(t) = 0$ represents the time at which the back of the car is at the garage door, so $s(0) = -4$ is the starting position of the car, 4 feet inside the garage.
a. Determine the velocity of the car when $s(t) = 0$.
b. Determine the velocity of the car when $s(t) = 14$.
156. The position of a hummingbird flying along a straight line in t seconds is given by $s(t) = 3t^3 - 7t$ meters.
a. Determine the velocity of the bird at $t = 1$ sec.
b. Determine the acceleration of the bird at $t = 1$ sec.
c. Determine the acceleration of the bird when the velocity equals 0.
157. A potato is launched vertically upward with an initial velocity of 100 ft/s from a potato gun at the top of an 85-foot-tall building. The distance in feet that the potato travels from the ground after t seconds is given by $s(t) = -16t^2 + 100t + 85$. . . .

MinerU

# 3.4 EXERCISES
For the following exercises, the given functions represent the position of a particle traveling along a horizontal line.
a. Find the velocity and acceleration functions. b. Determine the time intervals when the object is slowing down or speeding up.
150. $s(t) = 2t^3 - 3t^2 - 12t + 8$ 151. $s(t) = 2t^3 - 15t^2 + 36t - 10$ 152. $s(t) = \frac{t}{1+t^2}$
153. A rocket is fired vertically upward from the ground. The distance s in feet that the rocket travels from the ground after t seconds is given by $s(t) = -16t^2 + 560t$, .
a. Find the velocity of the rocket 3 seconds after being fired. b. Find the acceleration of the rocket 3 seconds after being fired.
154. A ball is thrown downward with a speed of 8 ft/ s from the top of a 64-foot-tall building. After t seconds, its height above the ground is given by $s(t) = -16t^2 - 8t + 64$. .
a. Determine how long it takes for the ball to hit the ground. b. Determine the velocity of the ball when it hits the ground.
155. The position function $s(t) = t^2 - 3t - 4$ represents the position of the back of a car backing out of a driveway and then driving in a straight line, where s is in feet and t is in seconds. In this case, $s(t) = 0$ represents the time at which the back of the car is at the garage door, so $s(0) = -4$ is the starting position of the car, 4 feet inside the garage.
a. Determine the velocity of the car when $s(t) = 0$. b. Determine the velocity of the car when $s(t) = 14$.
156. The position of a hummingbird flying along a straight line in t seconds is given by $s(t) = 3t^3 - 7t$ meters.
a. Determine the velocity of the bird at $t = 1$ sec. b. Determine the acceleration of the bird at $t = 1$ sec. c. Determine the acceleration of the bird when the velocity equals 0.
157. A potato is launched vertically upward with an initial velocity of 100 ft/s from a potato gun at the top of an 85-foot-tall building. The distance in feet that the potato travels from the ground after t seconds is given by $s(t) = -16t^2 + 100t + 85$. .
. . .

GOT-OCR 2.0

Chapter 3 | Derivatives 273 3.4 EXERCISES For the following exercises, the given functions represent the position of a particle traveling along a horizontal line. a. Find the velocity and acceleration functions. b. Determine the time intervals when the object is slowing down or speeding up. 150. s(t) = 2t3 −3t2 −12t + 8 151. s(t) = 2t3 −15t2 + 36t −10 152. s(t) = t 1 + t2 153. A rocket is fired vertically upward from the ground. The distance s in feet that the rocket travels from the ground after t seconds is given by s(t) = −16t2 + 560t. a. Find the velocity of the rocket 3 seconds after being fired. b. Find the acceleration of the rocket 3 seconds after being fired. 154. A ball is thrown downward with a speed of 8 ft/ s from the top of a 64-foot-tall building. After t seconds, its height above the ground is given by s(t) = −16t2 −8t + 64. a. Determine how long it takes for the ball to hit the ground. b. Determine the velocity of the ball when it hits the ground. 155. The position function s(t) = t2 −3t −4 represents the position of the back of a car backing out of a driveway and then driving in a straight line, where s is in feet and t is in seconds. In this case, s(t) = 0 represents the time at which the back of the car is at the garage door, so s(0) = −4 is the starting position of the car, 4 feet inside the garage. a. Determine the velocity of the car when s(t) = 0. b. Determine the velocity of the car when s(t) = 14. 156. The position of a hummingbird flying along a straight line in t seconds is given by s(t) = 3t3 −7t 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 4 4 2 2 2 2 2 2 2 2 2 0 1 1 1 1 1 1 1 1 3 4 4 4 4 4 4 4 4 3 4 4 4 4 4 4 4 4 2 3 3 3 3 3 3 3 3 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 0 2 2 2 2 2 2 2 2 2 5 5 5 5 5 5 5 5 5 1 1 1 1 1 1 1 1 1 5 5 5 5 5 5 5 5 5 0 0 0 0 0 0 0 0 0 5 5 5 5 5 5 5 5 5 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5 5 5 5 2 2 2 2 2 2 2 2 2 a. Use the graph of the position function to determine the time intervals when the velocity is positive, negative, or zero. b. Sketch the graph of the velocity function. c. Use the graph of the velocity function to determine the time intervals when the acceleration is positive, negative, or zero. d. Determine the time intervals when the object is speeding up or slowing down. . . .

Marker

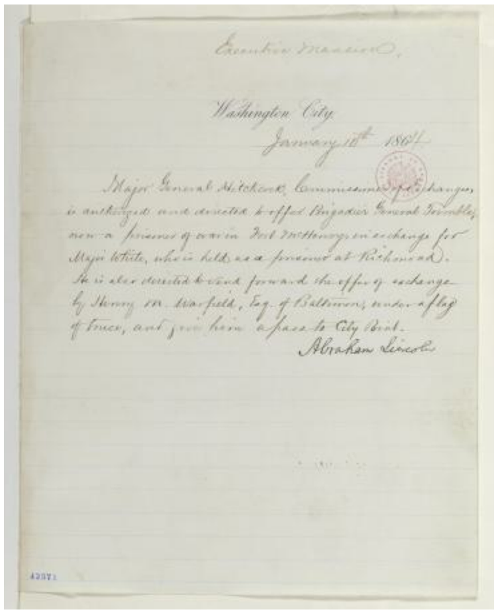
## **3.4 EXERCISES**
For the following exercises, the given functions represent the position of a particle traveling along a horizontal line.
- a. Find the velocity and acceleration functions. - b. Determine the time intervals when the object is slowing down or speeding up.

150. $s(t) = 2t^3 - 3t^2 - 12t + 8$

151. $s(t) = 2t^3 - 15t^2 + 36t - 10t$

152. $s(t) = \frac{t}{1 + t^2}$

153. A rocket is fired vertically upward from the ground. The distance *s* in feet that the rocket travels from the ground after *t* seconds is given by *s*(*t*) = −16*t* 2 + 560*t*.
- a. Find the velocity of the rocket 3 seconds after being fired. - b. Find the acceleration of the rocket 3 seconds after being fired.
154. A ball is thrown downward with a speed of 8 ft/ s from the top of a 64-foot-tall building. After *t* seconds, its height above the ground is given by *s*(*t*) = −16*t* 2 − 8*t* + 64.
- a. Determine how long it takes for the ball to hit the ground. - b. Determine the velocity of the ball when it hits the ground.
155. The position function *s*(*t*) = *t* 2 − 3*t* − 4 represents the position of the back of a car backing out of a driveway and then driving in a straight line, where *s* is in feet and *t* is in seconds. In this case, *s*(*t*) = 0 represents the time at which the back of the car is at the garage door, so *s*(0) = −4 is the starting position of the car, 4 feet inside the garage.
- a. Determine the velocity of the car when *s*(*t*) = 0. - b. Determine the velocity of the car when *s*(*t*) = 14.
156. The position of a hummingbird flying along a straight line in *t* seconds is given by *s*(*t*) = 3*t* 3 − 7*t* meters.
- a. Determine the velocity of the bird at *t* = 1 sec. - b. Determine the acceleration of the bird at *t* = 1 sec. - c. Determine the acceleration of the bird when the velocity equals 0.
. . .



Executive Mansion,

Washington City,

January 10th 1864

Major General Hitchcock, Commissioner of Exchanges, is authorized and directed to offer Brigadier General Trimble, now a prisoner of war in Fort McHenry, in exchange for Major White, who is held as a prisoner at Richmond. He is also directed to send forward the offer of exchange by Henry M. Warfield, Esq. of Baltimore, under a flag of truce, and give him a pass to City Point.

Abraham Lincoln

43571

olmOCR	MinerU	GOT-OCR 2.0	Marker
Executive Mansion, Washington City, January 15th, 1864 Major General Hitchcock, Commissioner of Exchanges, is authorized and directed to offer Brigadier General Trimble, now a prisoner of war in Fort McHenry, in exchange for Major White, who is held as a prisoner at Richmond. He is also directed to send forward the offer of exchange by Henry M. Warfield, Esq. of Baltimore, under a flag of truce, and give him a pass to City Point. Abraham Lincoln	*No text produced.*	43571 Bachington City January 10th 1864. Major General Architect, Commissioner of aivachangera is authorized and directed by ffeed Bngader General Trelmble, new a firemen of war in Fert nchery in exchange for Mayor White, who held a a firemen at Hillmannd. He is aker conducted by end forward the offer of exchange by Henry in. Warfield, Lag. of Balthmore, under a flag of three, and five him afaies to City Bink. Abraham Lincoln	necuhve Mansion Vastington amany layor Seneral Hitchcocks Commissioner of Cachanges, is anthonged and directed to offer Bingadier General Trin prisoner of war in Fort Inctienny, in exchange now w Major White, who is held as a preises at Richmond Ite is also directed to vand forwards the offer of exchange by Stenny in. Warfield, Eag. of Baltimore, under aflag 11 mice, and give him apass to tity Point. Abrakan Sincolus