

---

# Multimodal OCR: Parse Anything from Documents

---

Handong Zheng<sup>1\*</sup> Yumeng Li<sup>2\*</sup> Kaile Zhang<sup>1\*</sup> Liang Xin<sup>2\*</sup> Guangwei Zhao<sup>2</sup>  
Hao Liu<sup>2</sup> Jiayu Chen<sup>2</sup> Jie Lou<sup>2</sup> Jiyu Qiu<sup>2</sup> Qi Fu<sup>2</sup> Rui Yang<sup>2</sup>  
Shuo Jiang<sup>2</sup> Weijian Luo<sup>2</sup> Weijie Su<sup>2</sup> Weijun Zhang<sup>2</sup> Xingyu Zhu<sup>2</sup>  
Yabin Li<sup>2</sup> Yiwei Ma<sup>2</sup> Yu Chen<sup>2</sup> Zhaohui Yu<sup>2</sup> Guang Yang<sup>2†</sup>  
Colin Zhang<sup>2‡</sup> Lei Zhang<sup>2</sup> Yuliang Liu<sup>1†‡</sup> Xiang Bai<sup>1‡</sup>

<sup>1</sup>Huazhong University of Science and Technology    <sup>2</sup>hi lab, Xiaohongshu Inc

## Abstract

We present Multimodal OCR (MOCR), a document parsing paradigm that jointly parses text and graphics into unified textual representations. Unlike conventional OCR systems that focus on text recognition and leave graphical regions as cropped pixels, our method, termed dots.mocr, treats visual elements such as charts, diagrams, tables, and icons as first-class parsing targets, enabling systems to parse documents while preserving semantic relationships across elements. It offers several advantages: (1) it reconstructs both text and graphics as structured outputs, enabling more faithful document reconstruction; (2) it supports end-to-end training over heterogeneous document elements, allowing models to exploit semantic relations between textual and visual components; and (3) it converts previously discarded graphics into reusable code-level supervision, unlocking multimodal supervision embedded in existing documents. To make this paradigm practical at scale, we build a comprehensive data engine from PDFs, rendered webpages, and native SVG assets, and train a compact 3B-parameter model through staged pre-training and supervised fine-tuning. We evaluate dots.mocr from two perspectives: document parsing and structured graphics parsing. On document parsing benchmarks, it ranks second only to Gemini 3 Pro on our OCR Arena Elo leaderboard, surpasses existing open-source document parsing systems, and sets a new state of the art of 83.9 on olmOCR Bench. On structured graphics parsing, our model achieves higher reconstruction quality than Gemini 3 Pro across image-to-SVG benchmarks, demonstrating strong performance on charts, UI layouts, scientific figures, and chemical diagrams. These results show a scalable path toward building large-scale image-to-code corpora for multimodal pretraining. Code and models are publicly available at <https://github.com/rednote-hilab/dots.mocr>.

## 1 Introduction

In the era of large language and multimodal models, document parsing has become a core data engine for pretraining and retrieval because it determines how much reliable, structured supervision can be recovered from the massive volume of PDFs, scans, and screenshots that store real-world knowledge [10]. However, documents convey information not only through text but also through graphics such as charts, diagrams, flow charts, UI elements, and scientific illustrations. Existing document parsing pipelines remain largely text-centric: they focus on recognizing and organizing textual content while treating non-textual elements as figure regions that are simply cropped as

---

\*Core contribution

†Project leader

‡Corresponding author

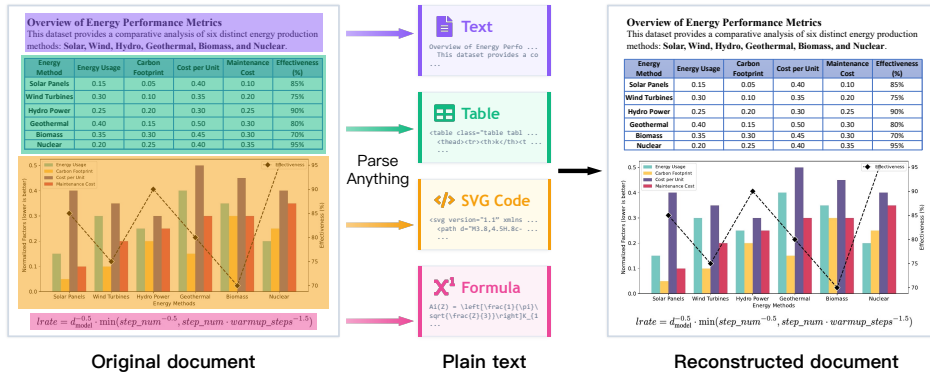


Figure 1: Overview of MOCR. Given a document image, MOCR parses anything on the page into unified, ordered textual representations, capturing both textual and visual structures and enabling faithful reconstruction of the original document.

pixels [26]. As a result, much of the structural and semantic information encoded in document graphics is discarded, making current document parsing inherently lossy and limiting the amount of supervision that can be extracted from documents [16, 25].

Recent advances in vision-language models make it increasingly feasible to recover structured representations from document visuals rather than preserving them as pixels. Beyond captioning, modern models show a growing capability to generate executable representations conditioned on images, enabling the reconstruction of underlying structures from visual inputs. Early work on translating UI screenshots into code (e.g., pix2code) explored this direction, and more recent approaches extend it to richer program spaces such as SVG, where images can be converted into renderable vector code [2, 30]. These developments suggest that document parsing can move beyond text extraction and instead aim to recover all information-bearing elements in documents as structured outputs.

Motivated by this observation, we introduce **Multimodal OCR (MOCR)**, a document parsing paradigm that aims to *parse anything* in documents, including text, layout structures, tables, and information-dense graphics such as charts, diagrams, icons, and UI components, as illustrated in Fig. 1. Unlike traditional OCR pipelines that primarily recover text while leaving graphical regions as raster crops (Fig. 3), MOCR treats both textual and visual elements as first-class parsing targets and converts them into reusable structured outputs. In particular, document graphics are represented as renderable SVG code together with textual content, allowing charts, diagrams, and other visual elements to be reconstructed as structured representations that can serve as reusable supervision for downstream reasoning and multimodal pretraining.

While MOCR provides a unified paradigm for parsing both textual and graphical elements, making it scalable remains challenging. First, supervision for graphics is scarce, as real documents rarely provide aligned program representations for visual elements. Second, renderable programs are inherently non-unique since different codes can produce visually identical outputs, requiring normalization and quality control during training. Third, the task demands precise visual grounding together with long-sequence structured generation, which is substantially more difficult than text-only OCR.

To address these challenges, we develop a scalable system, termed `dots.mocr`, trained with a large data engine spanning PDFs, rendered webpages, and native SVG graphics. Our training follows a staged recipe that combines large-scale OCR supervision with graphics-centric signals from naturally structured sources while applying normalization and quality control to align predicted code with faithful rendering. This design enables our method to generalize across both traditional document parsing and structured graphics reconstruction within a single unified architecture.

The main advantages of this work are summarized as follows:

- We introduce MOCR, a generalized OCR paradigm that elevates visual symbols to first-class parsing targets and recovers document graphics as reusable, renderable code rather than raster crops, unlocking a new source of structured supervision from existing documents.

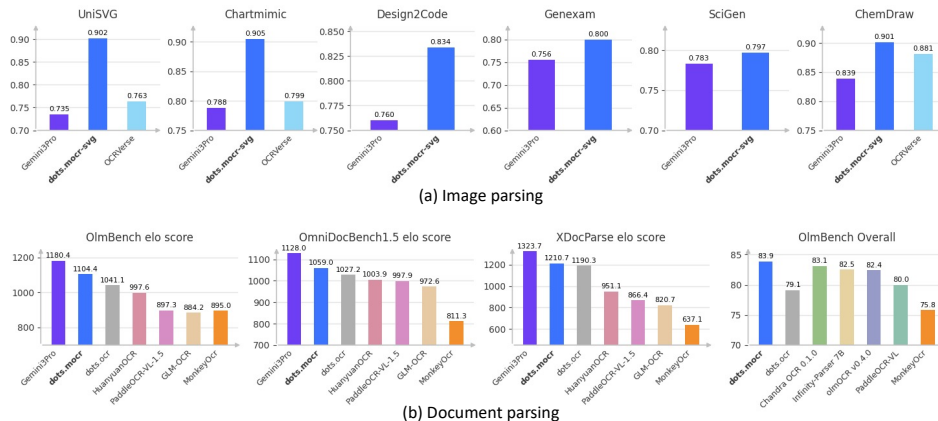


Figure 2: Overall performance comparison results. (a) reports metrics for image parsing tasks within documents. dots.mocr-svg are further enhanced with additional task-specific training for graphics parsing. (b) reports metrics for general document parsing tasks (olmOCR-Bench, OmniDocBench 1.5, and XDocParse).

- Our system, dots.mocr, proposes a unified learning formulation that makes it practical at scale under sparse and non-unique program supervision through normalization and training stabilization that align generated code with rendering fidelity, supported by a vision encoder trained entirely from scratch.
- dots.mocr demonstrates strong and balanced performance across document parsing and structured graphics reconstruction (Fig. 2), ranking second only to Gemini 3 Pro on the OCR Arena Elo leaderboard, setting a new state-of-the-art on olmOCR-Bench, surpassing Gemini 3 Pro on image-to-SVG benchmarks, and maintaining strong visual grounding and reasoning performance on OCRBench beyond parsing, all within a compact 3B-parameter model.

## 2 Related Work

### 2.1 Text Parsing

Text parsing methods have grown rapidly in recent years, aiming to extract and analyze textual content from diverse document formats, including PDFs, web pages, slides, spreadsheets, scanned documents, and scene-text images [11]. Existing approaches largely fall into three categories depending on whether and how they leverage vision–language models (VLMs). Traditional systems typically follow a multi-stage pipeline with layout analysis, detection, recognition, and reading-order prediction, as exemplified by pp-structurev3 [4], which offers modular designs and deployment-oriented integrations; however, they can accumulate errors across stages. A second line augments such pipelines with VLM components to strengthen semantic reasoning while retaining explicit structure, including MonkeyOCR [16], MinerU 2.5 [25], and PaddleOCR-VL [3]; these hybrids improve understanding in many settings yet remain primarily text-focused and inherit some pipeline complexity. Finally, end-to-end VLM-based models cast parsing as direct visual-to-text generation, such as DeepSeek-OCR [39], GOT-OCR [38], and OCRVerse [47], achieving strong cross-domain generalization via large-scale pretraining, while still facing challenges in maintaining faithful structure under dense layouts (e.g., tables and formulas).

### 2.2 Structured Graphics Parsing

Structured graphics parsing extends text parsing to the recovery of layout, geometry, and styling cues such as shapes, lines, and spatial relations, aiming to translate images into executable, renderable representations (for example, HTML, LaTeX, SVG, or Python) rather than character-level transcripts. Website and UI parsing exemplify this direction by converting screenshots into DOM-like structures

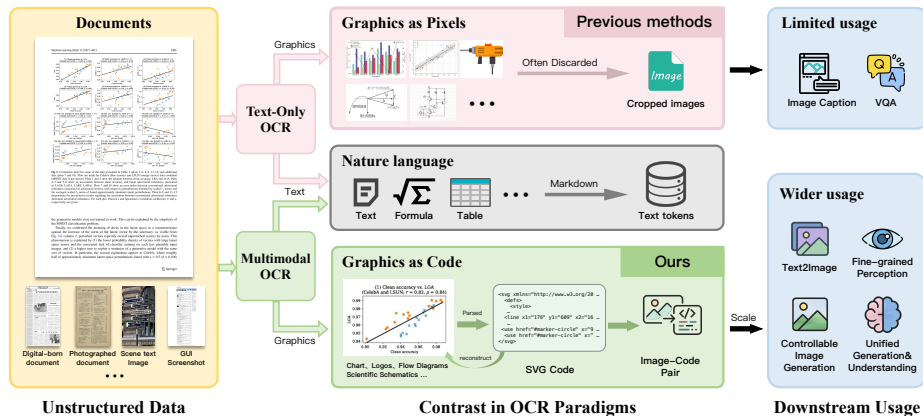


Figure 3: Comparison between traditional text-only OCR and MOCR paradigms. Traditional OCR treats graphics as pixels and often discards them, while MOCR parses graphics into structured code (e.g. SVG), enabling faithful reconstruction and broader downstream applications.

or front-end code: Pix2Struct [13] pretrains vision-to-text translation for simplified HTML from masked webpage images, Design2Code [31] benchmarks screenshot-to-implementation generation and highlights persistent fidelity gaps, and OmniParser [20] extracts UI elements directly from pixels. Beyond HTML, target languages are often domain-driven, with Plot2Code [41], ChartMimic [42], and ChartMaster [32] reconstructing charts via programmatic Python rendering, and ChemDraw-style settings mapping molecular diagrams to structured strings such as SMILES [45]. SVG has emerged as a particularly expressive target because it explicitly encodes geometry and style; recent methods translate images into SVG for icons and vector graphics, including StarVector [30], OmniSVG [43], and UniSVG [14]. While broader unification efforts such as OCRVerse [47] combine OCR, chart parsing, SVG reconstruction, web layout generation, and other structured targets within a single vision–language model through prompting, a persistent challenge is matching specialized systems on individual tasks while maintaining robust generalization to complex structured images.

In this context, we introduce MOCR to parse anything in documents, converting not only text but also charts, diagrams, UI elements, icons, and domain drawings into reusable, renderable representations instead of raster crops. MOCR aims to reframe document parsing as a scalable source of executable supervision for multimodal pretraining and retrieval, bridging text-focused parsers and task-specific graphic systems. An intuitive comparison among different systems is shown in Tab. 1 and Fig. 3.

### 3 Multimodal OCR

MOCR is designed by unifying page-level parsing tasks within a single model, including document parsing, webpage and UI parsing, scene-text parsing, and structured graphics parsing. This unification turns documents and screens into a richer data engine by recovering not only text but also visual symbols as reusable, renderable code (e.g., SVG) that is executable, editable, and compositional, enabling scalable supervision for pretraining and retrieval beyond raster crops.

#### 3.1 Task Definition

MOCR aims for the comprehensive parsing of document pages—including PDF renderings, digital scans, webpages, and scene-text images. Unlike traditional text-centric pipelines that treat non-textual elements as inert raster crops, MOCR treats both text and visual symbols as first-class parsing targets. This approach explicitly recovers information-dense graphics—such as charts, diagrams, icons, and schematics—into structured, reusable representations, thereby transforming static pixels into actionable data for downstream reasoning and multimodal training.

Given an input image  $I$ , the task is to generate an ordered sequence of parsed elements  $S$ :

$$S = [(B_1, c_1, p_1), \dots, (B_K, c_K, p_K)]. \quad (1)$$

Table 1: Comparison of different VLMs and OCR systems across multiple tasks. ✓ indicates Supported (good), △ indicates Supported but Underperforming, and ✗ indicates Not Supported.

Model Type	Inference Type	Model Name	Task					
			Layout-Det	Parsing	Spotting	VQA	IE	Graphics
Cascade Pipeline	Multi-Step	PaddleOCR-V5 [4]	✗	✗	✓	✗	✗	✗
		Marker-1.8.2	✓	✓	✗	✗	✗	
		PP-StructureV3 [4]	✓	✓	✗	✗	✗	
Specialized VLMs (Modular)	Multi-step	MonkeyOCR-pro-3B [16]	✓	✓	✗	✗	✗	✗
		MinerU2.5 [25]	✓	✓	✗	✗	✗	
		PaddleOCR-VL-1.5 [5]	✓	✓	✗	✗	✗	
		GLM-OCR [6]	✓	✓	✗	✗	✓	
Documents VLMs	One-Step	Text-Monkey [19]	✗	✗	✓	✓	✓	✗
		mPLUG-DocOwl2 [8]	✗	✓	✓	✓	✓	
General VLMs	One-Step	Qwen3-VL-235B-Instruct [1]	✓	✓	✓	✓	✓	△
		Gemini 3 pro [28]	✓	✓	✓	✓	✓	△
Specialized VLMs (End2End)	One-Step	dots.ocr [15]	✓	✓	✗	✗	✗	✗
		HunyuanOCR [33]	✗	✓	✓	✓	✓	✗
		DeepSeek-OCR2 [40]	✓	✓	✗	✗	✗	✗
		OCRVerse [47]	✗	✓	✗	✗	✗	△
		dots.mocr	✓	✓	✓	✓	✓	✓

Where each constituent element is defined by:  $\mathcal{B}_k$ , the spatial region or bounding box.  $c_k$ , the semantic category or element type.  $p_k$ , the associated payload. The sequence  $\mathbf{S}$  is generated following a human-centric reading order, allowing the model to encode structural hierarchies and logical relations implicitly through the generation sequence and specialized delimiters, rather than relying on an external relation module.

The payload  $p_k$  is a type-specific serialization of the content within region  $\mathcal{B}_k$ , determined by the semantic type  $c_k$ . For text-centric regions (e.g., text lines/blocks, tables, and formulas),  $p_k$  corresponds to their transcriptions in appropriate symbolic forms, such as plain text, table markup, or  $\LaTeX$ . For visual symbols that admit a concise, programmatic description—such as UI components, icons, and charts— $p_k$  is a renderable structured representation, i.e., image-to-SVG conversion. By parsing eligible graphics into SVG code, MOCR facilitates “render-and-reuse” workflows. Notably, complex real-world imagery or natural photographs, which lack a compact programmatic description, are retained as raster content. This strategic shift enables documents to contribute not only textual tokens but also granular, controllable structural supervision for the next generation of multimodal pretraining.

In the current release, MOCR is task-conditioned and does not yet produce a single one-pass output that simultaneously includes full-page document parsing and visual-symbol (e.g., SVG) parsing; instead, we obtain a complete multimodal parse by running page-level text parsing and region-level image-to-SVG decoding in separate passes.

### 3.2 Model Architecture

The architecture of our method adheres to the fundamental design principles established in previous work [15]. It comprises three primary components: a high-resolution vision encoder, a lightweight multimodal connector, and an autoregressive language model (LLM) decoder.

**High-Resolution Vision Encoder.** The vision encoder is a 1.2B-parameter backbone trained entirely from scratch, which ensures the encoder develops feature representations natively optimized for document parsing, enabling the joint modeling of dense text and geometry-sensitive visual symbols (e.g., charts, diagrams, and schematics). Architecturally, the encoder is engineered to ingest native high-resolution inputs of up to  $\sim 11\text{M}$  pixels. This high-capacity throughput is essential for preserving fine-grained details and maintaining long-range spatial coherence across a full page. Such resolution is critical not only for legibility in small-font text or dense layouts but also for the precise

perception of graphic primitives—such as chart markers and diagrammatic strokes—which must be accurately localized to be recovered as structured code.

**Structured Language Decoder.** For the autoregressive decoder, we use Qwen2.5-1.5B. The key consideration is the capacity and cost trade-off for unified MOCR parsing: models substantially smaller than 1.5B often struggle to simultaneously handle heterogeneous page content (text, layout structure, and visual symbols) and generate long, highly structured outputs such as SVG programs within a single autoregressive decoding process, while significantly larger decoders increase training and inference costs. Initializing from a base model (rather than a chat-specialized model) provides a neutral starting point for large-scale pretraining, where the model must learn non-natural, strongly structured target sequences and long-range dependencies as part of the parsing objective.

### 3.3 Training Recipe

Our training strategy is intentionally data-driven. Given the broad coverage of MOCR, our goal is not to introduce task-specific optimization heuristics, but to design an efficient curriculum that lowers learning difficulty, stabilizes multi-task joint training, and enables a single model to absorb heterogeneous supervision produced by our data engine.

We perform large-scale pretraining in three successive stages, each serving a distinct purpose. The first stage establishes a stable vision-language interface through general-purpose vision training so that the language model can reliably consume visual tokens and ground generation on visual inputs. The second stage conducts broad pretraining on a unified mixture of general vision data and text-only document parsing supervision, building strong text-centric parsing foundations while maintaining general visual robustness. The third stage shifts the mixture toward MOCR-specific targets by decreasing the proportion of general vision data and increasing the emphasis on multimodal document parsing, strengthening OCR-centric parsing together with visual-symbol parsing instantiated as image-to-SVG. Across all stages, we keep a single autoregressive objective, predicting structured parsing sequences conditioned on the input image and task instruction while controlling optimization stability via mixture reweighting and curriculum scheduling. We also progressively increase input resolution across stages to match the growing difficulty of dense page parsing and long structured generation.

After pretraining, we perform instruction tuning using a curated high-quality supervised set constructed by our data engine. Relative to pretraining, this stage prioritizes supervision reliability and task usability: we filter and refine examples to correct systematic errors, align output conventions, and improve end-to-end parsing fidelity across tasks. For visual-symbol parsing, instruction tuning is especially sensitive to target consistency, so SVG-specific handling (e.g., canonicalization, viewBox normalization, and complexity reduction) is treated as part of the data engine, while the training recipe focuses on integrating these refined signals into a stable multi-task SFT mixture. We release two checkpoints with the same pretraining: **dots.mocr** and **dots.mocr-svg**, where the latter increases the SVG share and up-weights harder SVG programs during SFT to better prioritize image-to-SVG parsing under the same parameter budget.

### 3.4 Data Engine

Training a single model for MOCR places unusually strict requirements on the training corpus. Beyond robustness to scripts, diverse layouts, and long-range reading structures, the model must also learn to parse visual symbols (such as charts, diagrams, icons, and schematics) into reusable structured representations rather than leaving them as raster crops. No existing dataset provides this coverage at sufficient scale and quality.

Our training corpus is built from four complementary sources: (i) PDF documents for text-language page parsing, (ii) web-derived pages rendered into images with aligned structural signals, (iii) native SVG assets for image-to-SVG supervision, and (iv) general-purpose data to maintain broad robustness and downstream usability. We apply lightweight quality control for pretraining to remove obvious noise while preserving diversity, and curate a smaller, higher-precision subset for instruction tuning with stricter verification and convention alignment.

**PDF documents.** We construct multilingual document parsing supervision from raw PDFs using dots.ocr as an auto-labeling engine, producing structured page transcriptions with layout regions and reading order. We curate the PDF pool via stratified sampling over language, domain, and layout

complexity (estimated by lightweight proxies such as block count, text density, and the presence of tables/formulas) to emphasize hard regimes. For instruction tuning, we further improve reliability through (i) verification with rule-based sanity checks and render-based comparison against the input page, and (ii) distillation that relabels or filters samples with stronger supervision to correct common errors.

**Webpages.** We crawl and render webpages into page images and convert them into the same MOCR parsing format as PDFs. This source broadens the distribution with naturally high-resolution and complex layouts, provides aligned structural signals from HTML/DOM to reduce label noise, and supplies abundant SVG-native icons, charts, and diagrams that further support visual-symbol parsing.

**SVG graphics.** A central goal of MOCR is to parse eligible graphics into reusable, renderable representations rather than keeping them as raster crops. Since many icons, charts, and UI graphics on the web are natively stored as SVG, we collect such assets from diverse sources and render them to construct image–SVG pairs. Our pipeline consists of two stages: cleaning and sampling. During cleaning, we use `svgo`<sup>4</sup> to remove irrelevant metadata, normalize numeric precision, and standardize code structure, followed by deduplication at both the code and image levels using textual matching and perceptual hashing (pHash) on rendered images. During sampling, we perform domain-level balancing to avoid over-representation from individual sources and apply complexity-aware sampling based on SVG program complexity to maintain a balanced mix of simple and complex graphics.

**General-purpose data.** We additionally include generic vision and OCR supervision (e.g., grounding and counting) to preserve broad capabilities alongside page-level parsing.

This data engine enables unified training over text parsing and visual-symbol parsing, converting previously raster-only graphics into reusable program supervision for MOCR.

### 3.5 Automated Evaluation via OCR Arena

Traditional metrics such as Word Error Rate (WER) or Normalized Edit Distance (NED), as well as structure-aware scores like TEDS [46] for tables and CDM [35] for formulas, often fail to reflect the true end-to-end quality of complex Markdown OCR outputs because they rely on rule-based matching to ground truth and are sensitive to non-unique but semantically equivalent serializations [7]. To address this limitation, we adopt an automated evaluation framework based on the LLM-as-a-Judge paradigm, referred to as *OCR Arena*. In this framework, a high-capacity vision-language model, such as Gemini 3 Flash, evaluates pairs of model outputs given the original document image and their generated Markdown transcriptions, judging which result better preserves fidelity, structure, and formatting or declaring a tie when both are comparable.

To ensure the integrity of our benchmarking and mitigate the well-documented issue of positional bias—where LLMs tend to favor the candidate presented first—we employ a rigorous symmetric evaluation protocol. Every pairwise comparison between Model *A* and Model *B* is conducted in two distinct trials: one where Model *A* is presented as the first candidate, and another where the presentation order is reversed. A model is credited with a victory only if the judge’s decision remains consistent across both trials. If the judge’s preference shifts solely based on the presentation order, or if the results are contradictory, the battle is categorized as inconsistent and treated as a tie. This dual-trial approach effectively filters out presentation artifacts and ensures that the final rankings reflect genuine model superiority.

To synthesize the outcomes of thousands of pairwise battles into a unified and interpretable leaderboard, we utilize the Elo rating system, a probabilistic ranking method originally developed for competitive games. Each model begins with an initial rating  $R$ , and for any given battle between Model *A* and Model *B*, the expected probability of Model *A* winning is calculated using the logistic curve:

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}} \quad (2)$$

Following the conclusion of a battle, the ratings are updated according to the actual outcome  $S_A$  (where  $S_A = 1$  for a win, 0.5 for a tie, and 0 for a loss) using the update rule:

$$R'_A = R_A + K \cdot (S_A - E_A) \quad (3)$$

---

<sup>4</sup><https://github.com/svg/svgo>

The factor  $K$  represents the sensitivity of the rating update, which we set to 32. This framework allows for a dynamic and scalable ranking that accounts for the relative strength of opponents, ensuring that victories against stronger models are weighted more heavily than those against weaker ones.

Given that the final Elo ratings can be influenced by the specific sequence in which battles are processed, we incorporate bootstrap resampling to enhance statistical robustness. We perform 1,000 iterations of the Elo calculation, with the entire battle history randomly shuffled in each round. The final reported Elo score for each model is the mean value derived from this distribution. This methodology guarantees that our leaderboard is stable and accurately represents the comparative performance of the OCR models across the entire evaluation dataset, providing a reliable measure of progress in high-fidelity document transcription.

## 4 Experiments

### 4.1 Document Parsing

Existing document parsing benchmarks often rely on strict matching rules over the predicted structured text, which can be sensitive to surface-form differences even when the underlying parsing is acceptable. To complement such evaluations, we adopt an Elo-style paired comparison protocol (introduced in the previous section), where model outputs are judged pairwise and aggregated into Elo ratings. In our setup, the Elo evaluation is conducted using Gemini 3 Flash as the judge, and all models in Tab. 2 are evaluated under the same protocol, making the Elo scores directly comparable.

Table 2: Elo comparison on olmOCR-Bench [29], OmniDocBench(v1.5) [26], and XDocParse [15]. All models are evaluated under the same Elo protocol with Gemini 3 Flash as judge, on benchmark-provided document images.

Models	olmOCR-Bench	OmniDocBench1.5	XDocParse	Average
Gemini 3 Pro [28]	<b>1180.4</b>	<b>1128.0</b>	<b>1323.7</b>	<b>1210.7</b>
dots.mocr	1104.4	1059.0	1210.7	1124.7
dots.ocr [15]	1041.1	1027.2	1190.3	1086.2
HunyuanOCR [33]	997.6	1003.9	951.1	984.2
PaddleOCR-VL-1.5 [5]	897.3	997.9	866.4	920.5
GLM-OCR [6]	884.2	972.6	820.7	892.5
MonkeyOCR-pro-3B [16]	895.0	811.3	637.1	781.1

Across all three benchmarks, dots.mocr achieves the strongest Elo performance among the open-source models listed in the table, indicating consistently strong text-language parsing quality under the same Elo protocol. Gemini 3 Pro ranks first in all three benchmarks in this comparison.

Table 3: Performance comparison on olmOCR-Bench.

Model	ArXiv	Old scans math	Tables	Old scans	Headers & footers	Multi column	Long tiny text	Base	Overall
MinerU 2.5.4 [25]	76.6	54.6	84.9	33.7	<b>96.6</b>	78.2	83.5	93.7	75.2±1.1
MonkeyOCR-pro-3B [16]	83.8	68.8	74.6	36.1	91.2	76.6	80.1	95.3	75.8±1.0
dots.ocr [15]	82.1	64.2	88.3	40.9	94.1	82.4	81.2	99.5	79.1±1.0
DeepSeek-OCR [39]	77.2	73.6	80.2	33.3	96.1	66.4	79.4	99.8	75.7±1.0
Nanonets-OCR2-3B [21]	75.4	46.1	86.8	40.9	32.1	81.9	<b>93.0</b>	99.6	69.5±1.1
PaddleOCR-VL [3]	85.7	71.0	84.1	37.8	97.0	79.9	85.7	98.5	80.0±1.0
Infinity-Parser 7B [34]	84.4	83.8	85.0	47.9	88.7	84.2	86.4	99.8	82.5±?
olmOCR v0.4.0 [29]	83.0	82.3	84.9	47.7	96.1	83.7	81.9	99.7	82.4±1.1
dots.mocr	<b>85.9</b>	<b>85.5</b>	<b>90.7</b>	<b>48.2</b>	94.0	<b>85.3</b>	81.6	99.7	<b>83.9±0.9</b>

We note that Elo ratings are not expected to numerically align with the strict-match metrics or leaderboard scores reported by existing benchmarks. First, our Elo protocol relies on pairwise judging rather than surface-form string matching, which mitigates known brittleness in rule-based evaluation (e.g., over-penalizing minor formatting or normalization differences) and can therefore shift absolute rankings. Second, the benchmark-provided image sets contain relatively fewer formula- and table-heavy pages; since Elo aggregates per-sample pairwise comparisons, these pages receive a proportionally smaller effective weight, consistent with our qualitative observations. In practice, models with lower Elo ratings often lose even on visually simple pages due to small but salient character-level mistakes (e.g., a single wrong symbol), which are consistently judged as worse in head-to-head comparisons. Overall, Elo provides complementary signals beyond strict matching by capturing error severity and robustness on common document pages.

To further analyze performance across document regimes, Tab. 3 reports a category breakdown on olmOCR-Bench. dots.mocr achieves the best overall score among all reported systems, and it also attains the highest scores on ArXiv, Old scans math, Tables, and Multi column. For other categories (e.g., Old scans, Headers & footers, Long tiny text, and Base), the best scores are achieved by other systems in the table, suggesting remaining headroom in these regimes.

## 4.2 Structured Graphics Parsing

We evaluate structured graphics parsing on a diverse set of benchmarks covering key visual domains: general vector graphics (UniSVG [14]); scientific charts (ChartMimic [42]); webpage and UI layouts (Design2Code [31]); exam-style diagrams (GenExam [37]); scientific figures (SciGen [17]); and chemistry structure diagrams (ChemDraw [45]). Together, these datasets span icons, charts, layouts, and domain-specific scientific drawings, enabling a broad evaluation of reconstruction quality.

For each benchmark, we use the original image as input, render the predicted structured code, and compute the ISVGEN score (from UniSVG [14]) between the rendered result and the original image. This render-and-compare protocol provides a unified reconstruction-based metric across all datasets in Tab. 4. We compare with an open-source baseline, OCRVerse, and a strong closed-source model, Gemini 3 Pro. OCRVerse generates different program formats depending on the task (e.g., SVG or Python), while Gemini 3 Pro and our models use SVG outputs; all methods are evaluated consistently through rendering and ISVGEN.

Table 4: ISVGEN reconstruction scores on UniSVG (low-level, high-level, overall) and downstream visual-language benchmarks (higher is better).

Methods	UniSVG <sup>Low-Level</sup>	UniSVG <sup>High-Level</sup>	UniSVG <sup>Score</sup>	ChartMimic	Design2Code	GenExam	SciGen	ChemDraw
OCRVerse [47]	0.632	0.852	0.763	0.799	-	-	-	0.881
Gemini 3 Pro [28]	0.563	0.850	0.735	0.788	0.760	0.756	0.783	0.839
dots.mocr	0.850	0.923	0.894	0.772	0.801	0.664	0.660	0.790
dots.mocr-svg	<b>0.860</b>	<b>0.931</b>	<b>0.902</b>	<b>0.905</b>	<b>0.834</b>	<b>0.800</b>	<b>0.797</b>	<b>0.901</b>

As shown in Tab. 4, dots.mocr-svg achieves the best overall performance across datasets, surpassing OCRVerse by +0.139 on UniSVG overall (0.902 vs. 0.763) and outperforming Gemini 3 Pro on all reported downstream benchmarks. The improvements are especially clear on structure-sensitive tasks such as ChartMimic and ChemDraw, while remaining strong on layouts and scientific figures. Given the 3B parameter scale, the base dots.mocr does not fully focus on visual-language parsing; dots.mocr-svg strengthens this ability by training with more visual-language data, and we will continue improving structured graphics understanding in future updates.

## 4.3 General VQA Evaluation

Table 5: Performance comparison on CharXiv, OCR reasoning, and a suite of document and vision-language benchmarks.

Model	CharXiv <sup>Descriptive</sup>	CharXiv <sup>Reasoning</sup>	OCR Reasoning	InfoVQA	DocVQA	ChartQA	OCRBench	AI2D	CountBenchQA	RefCOCO
Qwen3-VL-2B-Instruct [1]	62.3	26.8	-	72.4	93.3	-	85.8	76.9	88.4	-
Qwen3-VL-4B-Instruct [1]	76.2	39.7	-	80.3	95.3	-	88.1	84.1	84.9	-
dots.mocr	77.4	55.3	22.85	73.76	91.85	83.2	86.0	82.16	94.46	80.03

**General Capability.** Although dots.mocr is primarily designed for multimodal structured parsing, we further evaluate its general vision-language capability on a suite of benchmarks covering document understanding and broader multimodal reasoning (Tab. 5). Overall, dots.mocr remains highly competitive against a strong general-purpose baseline. In particular, it shows clear advantages on CharXiv [36] in both descriptive and reasoning settings, suggesting stronger fine-grained text-centric understanding and multimodal reasoning. On downstream document VQA and chart understanding tasks (e.g., OCR Reasoning [9], InfoVQA [24], DocVQA [23], and ChartQA [22]), dots.mocr achieves consistently strong results, indicating that unifying document parsing with broader VLM training does not compromise general-purpose performance. We also observe solid performance on OCRBench [18], AI2D [12], CountBenchQA [27], and RefCOCO [44], demonstrating that dots.mocr maintains broad visual grounding and reasoning abilities beyond parsing.

## 5 Qualitative examples

We present qualitative results to illustrate the structural understanding and reconstruction capabilities of `dots.mocr` and `dots.mocr-svg` across diverse visual domains. Instead of producing raster crops, the models generate structured outputs, including layout partitions and executable SVG code, enabling reusable and machine-readable representations. As shown in Fig. 4 and 5, `dots.mocr` performs robust layout analysis on heterogeneous documents, handling multilingual pages, complex multi-column layouts, dense tables, mathematical formulas, scanned materials, and handwritten notes. It also parses full-length web screenshots with preserved reading order and structured components, and recognizes scene text in real-world images, demonstrating strong generalization beyond traditional OCR settings.

Fig. 6–8 further show that `dots.mocr-svg` converts images into structured SVG code for vector-level reconstruction. The model supports basic icons, diverse statistical charts (e.g., bar, line, scatter, and composite charts), as well as complex cross-disciplinary illustrations in science and design, capturing geometric structure and semantic grouping in an editable and scalable representation.

As illustrated in Fig. 9, while retaining strong document parsing and SVG reconstruction capabilities, the model also maintains competitive general-purpose vision-language performance, producing coherent and context-aware responses across documents, charts, UI screenshots, and complex illustrations.

## 6 Discussion

Beyond the immediate benchmarks, we believe MOCR introduces a new direction for the broader community. By converting document graphics into image–code pairs, MOCR provides a scalable pipeline for constructing large multimodal pretraining corpora. Each chart parsed into SVG and each diagram converted into structured markup can form faithful image, code, and text triples, creating controllable and perturbable training data at a scale limited only by the number of available documents. Although this work instantiates graphics parsing as image-to-SVG, the MOCR paradigm is representation agnostic. Future extensions could target other program spaces such as TikZ for scientific figures, D3.js for interactive visualizations, CAD formats for engineering drawings, or domain-specific markup for chemical structures and circuit diagrams. Moreover, the ability to parse full webpages with diverse layouts, embedded graphics, and multilingual content substantially expands the pool of available training data beyond traditional PDF-centric corpora.

At the system level, MOCR also suggests new opportunities in data construction and evaluation. Our data engine shows that principled normalization, together with render based verification, can address the challenge of non-unique program targets. Scaling these quality control mechanisms through tighter render verification loops, reward model filtering, and self improving data curation offers a clear path to further improvements. Finally, our OCR Arena framework provides a practical alternative to brittle rule-based metrics, and similar judge-based evaluation protocols may become increasingly important as parsing outputs grow in structural complexity and representational diversity.

## 7 Conclusion

This work introduces Multimodal OCR (MOCR), a new document parsing paradigm that broadens document parsing from text extraction to a structured understanding of all information-bearing elements. Conventional systems typically stop at text and treat graphics, charts, diagrams, and icons as opaque pixel regions that are detected but not structurally interpreted. In contrast, the proposed paradigm treats every information-bearing element as a first-class parsing target and recovers visual symbols as reusable, renderable structured code. The key insight is that the richest supervision in many documents is often visual rather than textual; yet, it has historically been discarded by OCR pipelines. By recovering graphics as executable and re-renderable SVG programs, this work transforms static document pixels into structured supervision that can be reused for reasoning and learning. This capability not only broadens the scope of document parsing, but also increases the amount of usable supervision that can be extracted from large document corpora. More broadly, we hope this work helps shift document parsing from text-centric OCR toward document-native multimodal parsing that captures the full visual language of human knowledge.

## References

- [1] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.
- [2] Tony Beltramelli. pix2code: Generating code from a graphical user interface screenshot. In *Proceedings of the ACM SIGCHI symposium on engineering interactive computing systems (EICS)*, pages 1–6, 2018.
- [3] Cheng Cui, Ting Sun, Suyin Liang, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Xueqing Wang, Changda Zhou, Hongen Liu, Manhui Lin, et al. Paddleocr-vl: Boosting multilingual document parsing via a 0.9b ultra-compact vision-language model. *arXiv preprint arXiv:2510.14528*, 2025.
- [4] Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, et al. Paddleocr 3.0 technical report. *arXiv preprint arXiv:2507.05595*, 2025.
- [5] Cheng Cui, Ting Sun, Suyin Liang, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Xueqing Wang, Changda Zhou, Hongen Liu, Manhui Lin, et al. Paddleocr-vl-1.5: Towards a multi-task 0.9b vlm for robust in-the-wild document parsing. *arXiv preprint arXiv:2601.21957*, 2026.
- [6] Shuaiqi Duan, Yadong Xue, Weihang Wang, Zhe Su, Huan Liu, Sheng Yang, Guobing Gan, Guo Wang, Zihan Wang, Shengdong Yan, Dexin Jin, Yuxuan Zhang, Guohong Wen, Yanfeng Wang, Yutao Zhang, Xiaohan Zhang, Wenyi Hong, Yukuo Cen, Da Yin, Bin Chen, Wenmeng Yu, Xiaotao Gu, and Jie Tang. Glm-ocr technical report, 2026.
- [7] Pius Horn and Janis Keuper. Benchmarking document parsers on mathematical formula extraction from pdfs. *arXiv preprint arXiv:2512.09874*, 2025.
- [8] Anwen Hu, Haiyang Xu, Liang Zhang, Jiabo Ye, Ming Yan, Ji Zhang, Qin Jin, Fei Huang, and Jingren Zhou. mplug-docowl2: High-resolution compressing for ocr-free multi-page document understanding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5817–5834, 2025.
- [9] Mingxin Huang, Yongxin Shi, Dezhi Peng, Songxuan Lai, Zecheng Xie, and Lianwen Jin. Ocr-reasoning benchmark: Unveiling the true capabilities of mlms in complex text-rich image reasoning. *arXiv preprint arXiv:2505.17163*, 2025.
- [10] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM international conference on multimedia (ACMMM)*, pages 4083–4091, 2022.
- [11] Wenjun Ke, Yifan Zheng, Yining Li, Hengyuan Xu, Dong Nie, Peng Wang, and Yao He. Large language models in document intelligence: A comprehensive survey, recent advances, challenges, and future trends. *ACM Transactions on Information Systems (TOIS)*, 44(1):1–64, 2025.
- [12] Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *European conference on computer vision (ECCV)*, pages 235–251, 2016.
- [13] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. pages 18893–18912, 2023.
- [14] Jinke Li, Jiarui Yu, Chenxing Wei, Hande Dong, Qiang Lin, Liangjing Yang, Zhicai Wang, and Yanbin Hao. Unisvg: A unified dataset for vector graphic understanding and generation with multimodal large language models. In *Proceedings of the 33rd ACM International Conference on Multimedia (ACMMM)*, pages 13156–13163, 2025.
- [15] Yumeng Li, Guang Yang, Hao Liu, Bowen Wang, and Colin Zhang. dots.ocr: Multilingual document layout parsing in a single vision-language model. *arXiv preprint arXiv:2512.02498*, 2025.
- [16] Zhang Li, Yuliang Liu, Qiang Liu, Zhiyin Ma, Ziyang Zhang, Shuo Zhang, Zidun Guo, Jiarui Zhang, Xinyu Wang, and Xiang Bai. Monkeyocr: Document parsing with a structure-recognition-relation triplet paradigm. *arXiv preprint arXiv:2506.05218*, 2025.
- [17] Honglin Lin, Chonghan Qin, Zheng Liu, Qizhi Pei, Yu Li, Zhanping Zhong, Xin Gao, Yanfeng Wang, Conghui He, and Lijun Wu. Scientific image synthesis: Benchmarking, methodologies, and downstream utility. *arXiv preprint arXiv:2601.17027*, 2026.

- [18] Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences (SCIS)*, 67(12):220102, 2024.
- [19] Yuliang Liu, Biao Yang, Qiang Liu, Zhang Li, Zhiyin Ma, Shuo Zhang, and Xiang Bai. Textmonkey: An ocr-free large multimodal model for understanding document. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2026.
- [20] Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*, 2024.
- [21] Souvik Mandal, Ashish Talewar, Siddhant Thakuria, Paras Ahuja, and Prathamesh Juvatkar. Nanonets-ocr2: A model for transforming documents into structured markdown with intelligent content recognition and semantic tagging, 2025.
- [22] Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics (ACL)*, pages 2263–2279, 2022.
- [23] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision (WACV)*, pages 2200–2209, 2021.
- [24] Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. Infographicvqa. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1697–1706, 2022.
- [25] Junbo Niu, Zheng Liu, Zhuangcheng Gu, Bin Wang, Linke Ouyang, Zhiyuan Zhao, Tao Chu, Tianyao He, Fan Wu, Qintong Zhang, et al. Mineru2. 5: A decoupled vision-language model for efficient high-resolution document parsing. *arXiv preprint arXiv:2509.22186*, 2025.
- [26] Linke Ouyang, Yuan Qu, Hongbin Zhou, Jiawei Zhu, Rui Zhang, Qunshu Lin, Bin Wang, Zhiyuan Zhao, Man Jiang, Xiaomeng Zhao, et al. Omnidocbench: Benchmarking diverse pdf document parsing with comprehensive annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24838–24848, 2025.
- [27] Roni Paiss, Ariel Ephrat, Omer Tov, Shiran Zada, Inbar Mosseri, Michal Irani, and Tali Dekel. Teaching clip to count to ten. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3170–3180, 2023.
- [28] Sundar Pichai, Demis Hassabis, and Koray Kavukcuoglu. A new era of intelligence with gemini 3, 2025.
- [29] Jake Poznanski, Aman Rangapur, Jon Borchardt, Jason Dunkelberger, Regan Huff, Daniel Lin, Christopher Wilhelm, Kyle Lo, and Luca Soldaini. olmocr: Unlocking trillions of tokens in pdfs with vision language models. *arXiv preprint arXiv:2502.18443*, 2025.
- [30] Juan A Rodriguez, Abhay Puri, Shubham Agarwal, Issam H Laradji, Pau Rodriguez, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. Starvector: Generating scalable vector graphics code from images and text. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 16175–16186, 2025.
- [31] Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. Design2code: Benchmarking multimodal code generation for automated front-end engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL)*, pages 3956–3974, 2025.
- [32] Wentao Tan, Qiong Cao, Chao Xue, Yibing Zhan, Changxing Ding, and Xiaodong He. Chartmaster: Advancing chart-to-code generation with real-world charts and chart similarity reinforcement learning. *arXiv preprint arXiv:2508.17608*, 2025.
- [33] Hunyuan Vision Team, Pengyuan Lyu, Xingyu Wan, Gengluo Li, Shangpin Peng, Weinong Wang, Liang Wu, Huawei Shen, Yu Zhou, Canhui Tang, et al. Hunyuanocr technical report. *arXiv preprint arXiv:2511.19575*, 2025.
- [34] Baode Wang, Biao Wu, Weizhen Li, Meng Fang, Zuming Huang, Jun Huang, Haozhe Wang, Yanjie Liang, Ling Chen, Wei Chu, et al. Infinity parser: Layout-aware reinforcement learning for scanned document parsing. *arXiv preprint arXiv:2506.03197*, 2025.

- [35] Bin Wang, Fan Wu, Linke Ouyang, Zhuangcheng Gu, Rui Zhang, Renqiu Xia, Botian Shi, Bo Zhang, and Conghui He. Image over text: Transforming formula recognition evaluation with character detection matching. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 19681–19690, 2025.
- [36] Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, et al. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *Advances in Neural Information Processing Systems (Neurips)*, 37:113569–113697, 2024.
- [37] Zhaokai Wang, Penghao Yin, Xiangyu Zhao, Changyao Tian, Yu Qiao, Wenhai Wang, Jifeng Dai, and Gen Luo. Genexam: A multidisciplinary text-to-image exam. *arXiv preprint arXiv:2509.14232*, 2025.
- [38] 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.
- [39] Haoran Wei, Yaofeng Sun, and Yukun Li. Deepseek-ocr: Contexts optical compression. *arXiv preprint arXiv:2510.18234*, 2025.
- [40] Haoran Wei, Yaofeng Sun, and Yukun Li. Deepseek-ocr 2: Visual causal flow. *arXiv preprint arXiv:2601.20552*, 2026.
- [41] Chengyue Wu, Zhixuan Liang, Yixiao Ge, Qiushan Guo, Zeyu Lu, Jiahao Wang, Ying Shan, and Ping Luo. Plot2code: A comprehensive benchmark for evaluating multi-modal large language models in code generation from scientific plots. In *Findings of the Association for Computational Linguistics (NAACL)*, pages 3006–3028, 2025.
- [42] Cheng Yang, Chufan Shi, Yaxin Liu, Bo Shui, Junjie Wang, Mohan Jing, Linran Xu, Xinyu Zhu, Siheng Li, Yuxiang Zhang, et al. Chartmimic: Evaluating lmm’s cross-modal reasoning capability via chart-to-code generation. *arXiv preprint arXiv:2406.09961*, 2024.
- [43] Yiying Yang, Wei Cheng, Sijin Chen, Xianfang Zeng, Fukun Yin, Jiayu Zhang, Liao Wang, Gang Yu, Xingjun Ma, and Yu-Gang Jiang. Omnisvg: A unified scalable vector graphics generation model. *arXiv preprint arXiv:2504.06263*, 2025.
- [44] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *European conference on computer vision (ECCV)*, pages 69–85, 2016.
- [45] Xuanle Zhao, Deyang Jiang, Zhixiong Zeng, Lei Chen, Haibo Qiu, Jing Huang, Yufeng Zhong, Liming Zheng, Yilin Cao, and Lin Ma. Vincicoder: Unifying multimodal code generation via coarse-to-fine visual reinforcement learning. *arXiv preprint arXiv:2511.00391*, 2025.
- [46] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: Data, model, and evaluation. In *European conference on computer vision (ECCV)*, pages 564–580, 2020.
- [47] Yufeng Zhong, Lei Chen, Xuanle Zhao, Wenkang Han, Liming Zheng, Jing Huang, Deyang Jiang, Yilin Cao, Lin Ma, and Zhixiong Zeng. Ocrverse: Towards holistic ocr in end-to-end vision-language models. *arXiv preprint arXiv:2601.21639*, 2026.

# A Appendix

## A.1 Other Benchmarks

Table 6: Comparison of general-purpose and specialized VLMs on OmniDocBench(v1.5) and pdf-parse-bench. Lower is better for edit metrics.

Model Type	Methods	Size	OmniDocBench(v1.5)TextEdit ↓	OmniDocBench(v1.5)Read OrderEdit ↓	pdf-parse-bench
GeneralVLMs	Gemini 2.5 Pro	-	0.075	0.097	9.06
	Qwen3-VL-235B-A22B-Instruct [1]	235B	0.069	0.068	9.71
	Gemini 3 Pro [28]	-	0.066	0.079	9.68
SpecializedVLMs	Mistral OCR	-	0.164	0.144	8.84
	DeepSeek-OCR [39]	3B	0.073	0.086	8.26
	MonkeyOCR-3B [16]	3B	0.075	0.129	9.27
	OCRVerse [47]	4B	0.058	0.071	-
	MonkeyOCR-pro-3B [16]	3B	0.075	0.128	-
	MinerU2.5 [25]	1.2B	0.047	0.044	-
	PaddleOCR-VL [3]	0.9B	0.035	0.043	9.51
	HunyuanOCR [33]	0.9B	0.042	-	-
	PaddleOCR-VL1.5 [5]	0.9B	0.035	0.042	-
	GLMOCR [6]	0.9B	0.040	0.043	-
	dots.ocr [15]	3B	0.048	0.053	9.29
	dots.mocr	3B	<b>0.031</b>	<b>0.029</b>	9.54

Metrics are collected from OmniDocBench v1.5 [26] and reported numbers in corresponding model publications. Results on pdf-parse-bench [7] are reproduced by Qwen3-VL-235B-A22B-Instruct under a unified evaluation setup. We omit OmniDocBench v1.5 Formula and Table metrics because they are highly sensitive to detection rules and ground-truth matching protocols, which may introduce non-negligible variance across implementations.

Overall, dots.mocr achieves the strongest performance on OmniDocBench v1.5 text transcription and reading-order metrics (TextEdit and ReadOrderEdit), indicating robust document parsing and ordering accuracy. In addition, we observe clear improvements over dots.ocr on formula-intensive pages, suggesting better recognition of mathematical content. In future work, we will prioritize boosting formula and table recognition performance via stronger high-quality supervision, targeted data scaling, and task-specific training that directly improves end-to-end recognition accuracy.

## A.2 Qualitative examples

This section presents qualitative examples of the outputs produced by dots.mocr and dots.mocr-svg, shown in Figures 4–9.

## A.3 OCR Arena

We adopt a high-capacity vision-language model (VLM) as an impartial judge (e.g., Gemini 3 Flash) to compare two Markdown OCR outputs conditioned on the same source document image. The judging prompt is shown in Figure 10 and some judging results are provided in Figure 11.

doc

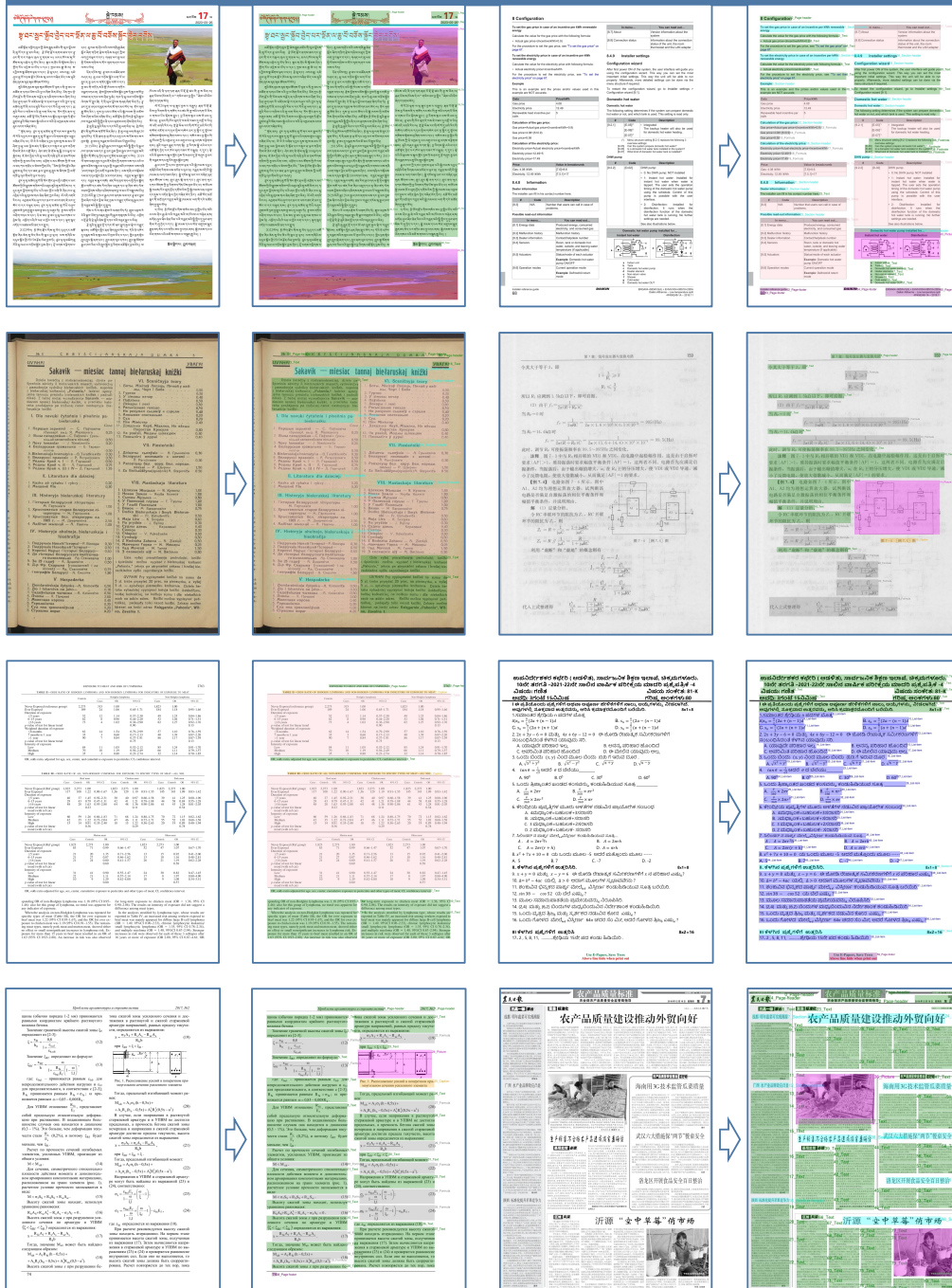


Figure 4: Qualitative layout analysis results of dots.mocr on heterogeneous documents. The model identifies and partitions structural elements such as titles, paragraphs, multi-column regions, dense tables, mathematical formulas, scanned text, and handwritten content, demonstrating robust document-level structural understanding across diverse formats and languages.

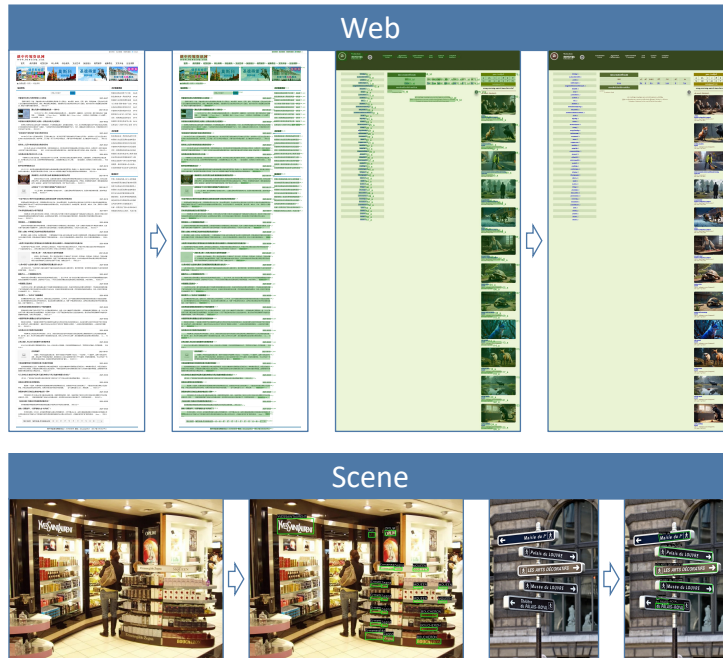


Figure 5: Parsing results of dots.mocr on full-length web screenshots and real-world scene images. The model preserves global reading order and structured components in long webpage layouts, while accurately recognizing and organizing scene text in complex real-world environments, highlighting generalization beyond conventional OCR benchmarks.

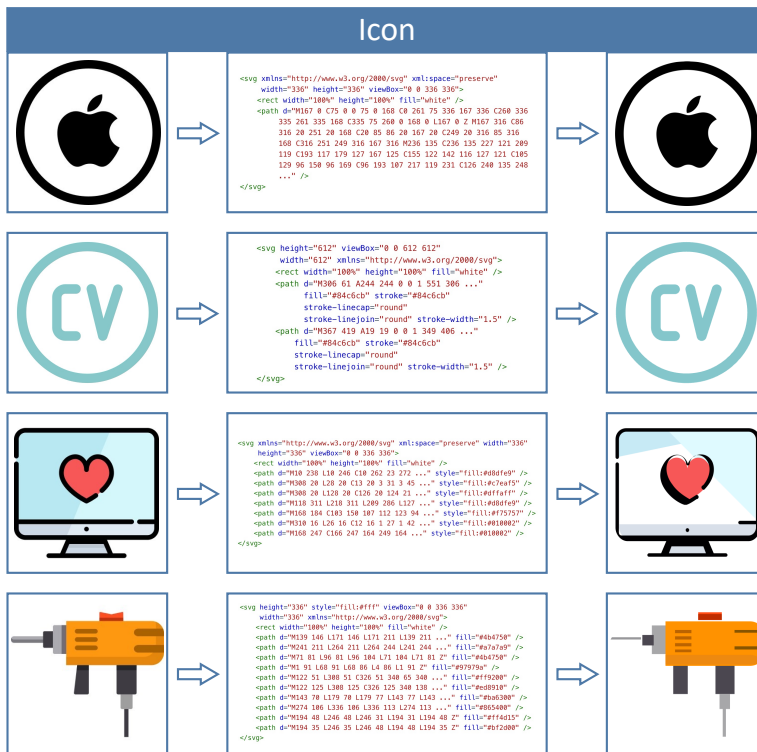


Figure 6: SVG parsing outputs and rendered reconstructions produced by dots.mocr-svg for icon images. The model converts raster icons into concise and executable SVG code, capturing geometric primitives, hierarchical grouping, and spatial relationships for scalable vector-level reconstruction.



Figure 7: Rendered SVG reconstruction results of dots.mocr-svg for diverse statistical charts, including bar, line, scatter, and composite visualizations. The model recovers chart structure, axes, legends, data encodings, and semantic grouping through executable SVG representations, enabling editable and reusable vector outputs.

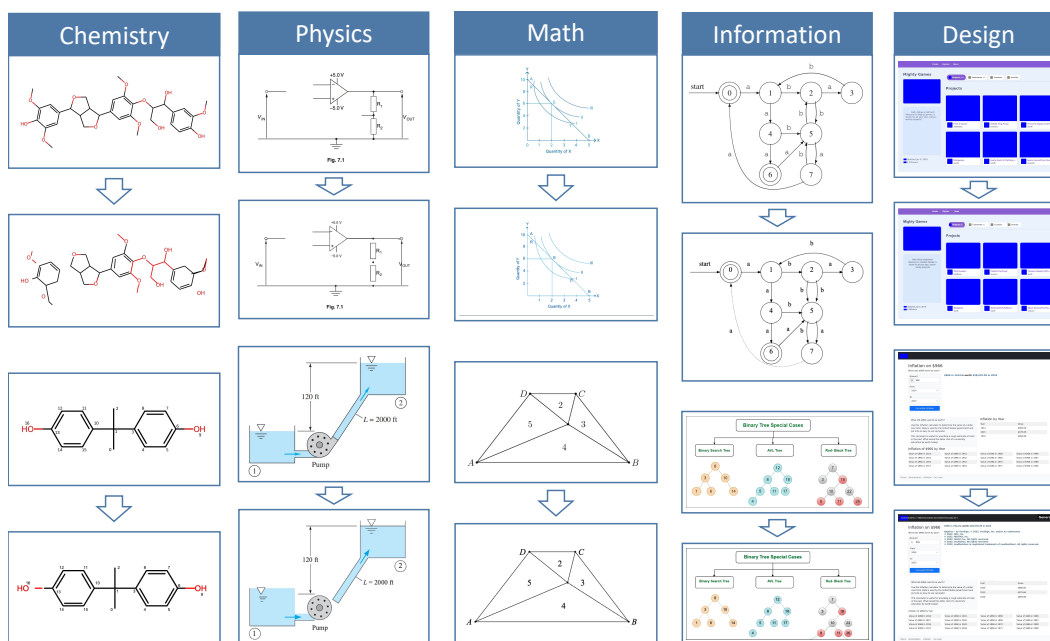
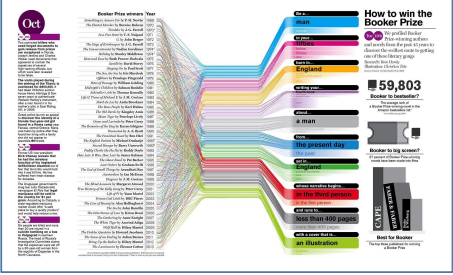


Figure 8: Rendered SVG reconstruction results of dots.mocr-svg for complex cross-disciplinary illustrations in science and design. The model captures intricate geometric structures, semantic components, and hierarchical organization, demonstrating its ability to generalize beyond standard charts to domain-specific structured graphics.

## General



**Booker Prize winners** Year  
1947: Goldsmith, *The Burmese Days*  
1948: Burgess, *The End of the World*  
1949: Faulkner, *The Sound and the Fury*  
1950: Woolf, *To the Lighthouse*  
1951: Hemingway, *A Farewell to Arms*  
1952: Faulkner, *As I Lay Dying*  
1953: Faulkner, *Go Down, Moses*  
1954: Faulkner, *Invisible Man*  
1955: Faulkner, *The Sound and the Fury*  
1956: Faulkner, *The Sound and the Fury*  
1957: Faulkner, *The Sound and the Fury*  
1958: Faulkner, *The Sound and the Fury*  
1959: Faulkner, *The Sound and the Fury*  
1960: Faulkner, *The Sound and the Fury*  
1961: Faulkner, *The Sound and the Fury*  
1962: Faulkner, *The Sound and the Fury*  
1963: Faulkner, *The Sound and the Fury*  
1964: Faulkner, *The Sound and the Fury*  
1965: Faulkner, *The Sound and the Fury*  
1966: Faulkner, *The Sound and the Fury*  
1967: Faulkner, *The Sound and the Fury*  
1968: Faulkner, *The Sound and the Fury*  
1969: Faulkner, *The Sound and the Fury*  
1970: Faulkner, *The Sound and the Fury*  
1971: Faulkner, *The Sound and the Fury*  
1972: Faulkner, *The Sound and the Fury*  
1973: Faulkner, *The Sound and the Fury*  
1974: Faulkner, *The Sound and the Fury*  
1975: Faulkner, *The Sound and the Fury*  
1976: Faulkner, *The Sound and the Fury*  
1977: Faulkner, *The Sound and the Fury*  
1978: Faulkner, *The Sound and the Fury*  
1979: Faulkner, *The Sound and the Fury*  
1980: Faulkner, *The Sound and the Fury*  
1981: Faulkner, *The Sound and the Fury*  
1982: Faulkner, *The Sound and the Fury*  
1983: Faulkner, *The Sound and the Fury*  
1984: Faulkner, *The Sound and the Fury*  
1985: Faulkner, *The Sound and the Fury*  
1986: Faulkner, *The Sound and the Fury*  
1987: Faulkner, *The Sound and the Fury*  
1988: Faulkner, *The Sound and the Fury*  
1989: Faulkner, *The Sound and the Fury*  
1990: Faulkner, *The Sound and the Fury*  
1991: Faulkner, *The Sound and the Fury*  
1992: Faulkner, *The Sound and the Fury*  
1993: Faulkner, *The Sound and the Fury*  
1994: Faulkner, *The Sound and the Fury*  
1995: Faulkner, *The Sound and the Fury*  
1996: Faulkner, *The Sound and the Fury*  
1997: Faulkner, *The Sound and the Fury*  
1998: Faulkner, *The Sound and the Fury*  
1999: Faulkner, *The Sound and the Fury*  
2000: Faulkner, *The Sound and the Fury*  
2001: Faulkner, *The Sound and the Fury*  
2002: Faulkner, *The Sound and the Fury*  
2003: Faulkner, *The Sound and the Fury*  
2004: Faulkner, *The Sound and the Fury*  
2005: Faulkner, *The Sound and the Fury*  
2006: Faulkner, *The Sound and the Fury*  
2007: Faulkner, *The Sound and the Fury*  
2008: Faulkner, *The Sound and the Fury*  
2009: Faulkner, *The Sound and the Fury*  
2010: Faulkner, *The Sound and the Fury*  
2011: Faulkner, *The Sound and the Fury*  
2012: Faulkner, *The Sound and the Fury*  
2013: Faulkner, *The Sound and the Fury*  
2014: Faulkner, *The Sound and the Fury*  
2015: Faulkner, *The Sound and the Fury*  
2016: Faulkner, *The Sound and the Fury*  
2017: Faulkner, *The Sound and the Fury*  
2018: Faulkner, *The Sound and the Fury*  
2019: Faulkner, *The Sound and the Fury*  
2020: Faulkner, *The Sound and the Fury*  
2021: Faulkner, *The Sound and the Fury*  
2022: Faulkner, *The Sound and the Fury*  
2023: Faulkner, *The Sound and the Fury*

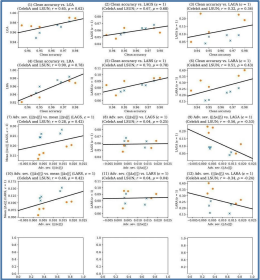
How does the narrative begin in "The Luminaries" by Eleanor Catton?

To determine how the narrative begins in "The Luminaries" by Eleanor Catton, we follow these steps:

1. Locate "The Luminaries" in the list of Booker Prize winners...
2. Follow the flow of the infographic to find the category ...
3. The section "whose narrative begins..." has the category "in the ..."

Thus, the narrative begins in "The Luminaries" by Eleanor Catton in the third person.

`\boxed{in the third person}`




Across panels 1-12 plotting against clean accuracy, which variable appears most positively correlated with clean accuracy?

To determine which variable appears most positively correlated with clean accuracy across panels 1-12, we analyze the correlation coefficients ( $r$ ) provided in each panel's title:

1. Panel (1): LGA vs. Clean accuracy, ( $r = 0.65$ ) ...
12. Panel (12): Adv. sev. ( $|\Delta x|_{2^s}$ ) vs. LARA ( $\epsilon = 1$ ), ( $r = -0.34$ )

We compare the positive correlation coefficients: 0.65, 0.67, 0.90, 0.70, 0.51, 0.46. The highest value is 0.90, which corresponds to LRA in Panel (4).

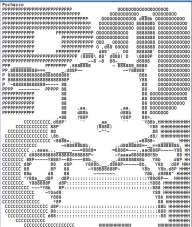
`\boxed{\text{LRA}}`



What is the wrong answer 2?


To determine the wrong answer 2, we look at the section labeled "Wrong Answer 2" in the image. The text there is "on".

`\boxed{on}`




Which of the following options best matches the image? Here's some options: Sesame Street, Pochacco, My Melody, Albert Einstein. Directly output the results without any additional explanation.

`Pochacco`



从图中提取: 来样日期, Add, NO., 样品来源, 样品名称, E-mail, 检验类别, 样品款号, 受检单位名称, Fax, 签发报告, Tel, 并按json格式返回

`[{"来样日期": "2012/10/23", "Add": "No.3 Renmin Rd Nantong.Jiangsu", "NO.": "2008100779C", "样品来源": "自送", "样品名称": "大提花漂白面料", "E-mail": "tsrmlfy518@sina.com", "检验类别": "委托", "样品款号": "/", "受检单位名称": "南通金赛岚纺织品有限公司", "Fax": "0513-81501382 85582799", "签发报告": "___", "Tel": "0513-85589758 85582788"}]`



How many Person are there in the image? To ensure accuracy, first detect their points using json format, then give the total number. Please represent the coordinates as normalized pixel coordinates from 0 to 1.

`[{"point": [0.491, 0.255], "label": "Person"},..., {"point": [0.024, 0.875], "label": "Person"}]`

5

Figure 9: General-purpose visual question answering results of dots.mocr. While retaining strong document parsing and SVG reconstruction capabilities, the model produces coherent and context-aware responses across documents, charts, UI screenshots, and complex illustrations, indicating competitive performance in broader vision-language tasks.

You are an expert in evaluating OCR content accuracy. Please compare the model outputs with the original image, focusing heavily on **content accuracy** while ignoring formatting and layout differences.

[Evaluation Focus – Focus ONLY on Content Accuracy]

- Text Accuracy**:
  - Typos: Character recognition errors (e.g., "test" recognized as "tost").
  - Omissions: Missing characters or words present in the original text.
  - Hallucinations: Adding characters that do not exist in the original text.
- Table Accuracy**:
  - Correctness of data and text within the table.
  - Completeness of cell content.
  - Correct row/column alignment.
- Formula Accuracy** (Evaluate based on):
  - **Correctness**: Are mathematical symbols, variables, and operators preserved accurately?
  - **Completeness**: Are all parts of the formula present without omission?
  - **Semantic Equivalence**: Does the extracted formula convey the exact same mathematical meaning?

[Tie Judgment Criteria – Important]

- You must judge as a **tie** in the following cases:
- Text content is identical, differing only in Markdown formatting.
  - Table data is identical, differing only in Markdown table syntax.
  - Formula content is semantically equivalent, differing only in LaTeX representation.
  - Both models correctly identified the core content; minor differences do not affect information retrieval.
  - Both models share the same minor errors or are both perfect.
  - **Image/Figure processing differs** (one extracts text, one gives bbox, one ignores it), but the main text is accurate.

[Items to Ignore – Do NOT factor into scoring]

- Markdown formatting differences (e.g., # Header vs ## Header, \* vs - for lists).
- Layout and typesetting differences (newlines, indentation, alignment).
- Recognition differences in non-body text like Headers, Footers, and Page Numbers.
- Text wrapping and paragraph segmentation nuances.
- Table border styles (e.g., |---|---| vs |:---|:|).
- Different but equivalent LaTeX representations for formulas.
- **Image/Figure Processing Differences (ABSOLUTELY IGNORE)**:
  - How the model parses image/figure regions is **completely excluded** from the scoring standard.
  - Whether it parses as a figure field, outputs bbox coordinates, extracts text inside the image, provides a caption, describes the image content, or **completely ignores/skips** the image, these are all considered equivalent.
  - Do NOT declare a winner based on image handling.

[Model 1 Output]:

```
markdown
{c1_text}
```

[Model 2 Output]:

```
markdown
{c2_text}
```

[Evaluation Process]

- Carefully compare the text content against the original image.
- Identify errors, omissions, or additions in text recognition for both models.
- Check the accuracy of table data.
- Evaluate the correctness, completeness, and semantic equivalence of mathematical formulas.
- Ignore image regions**: Confirm that differences in image/figure parsing are not used for scoring.
- Important**: If the substance is the same and only the format differs, judge as a tie.
- Only declare a winner if there is a significant difference in **content accuracy**.

[Examples of Ties]

- Model 1: "# Title", Model 2: "## Title" (Same content, different level).
- Model 1: "\* Item", Model 2: "- Item" (Same content, different bullet).
- Formula: Model 1 "\$x^2\$", Model 2 "\$x\*x\$" (Different LaTeX, same meaning).
- Table data is identical, but column alignment syntax differs.
- Identification is identical, but one model parsed the footer while the other didn't (Judge as Tie).
- **Image handling**: Model 1 outputs an image bbox, Model 2 outputs an image description, Model 3 ignores the image. As long as the main text is accurate, this is a **Tie**.

[Output Requirement] Please strictly return the result in the following JSON format:

```
{{"winner": "tie", "reason": "Detailed explanation of the judgment, specifically noting the logic for a tie"}}
```

The value of "winner" must be one of:

- "1": Model 1 is clearly better in content accuracy.
- "2": Model 2 is clearly better in content accuracy.
- "tie": Both models perform equally in content accuracy (including cases of identical content but different formatting/image handling).

In the "reason" field, specifically explain:

- If a tie: Explain the consistency of the content and explicitly mention which formatting or image handling differences were ignored.
- If a winner: Specifically point out the accuracy differences (typos, missing words, table/formula errors).
- **Note**: It is better to judge a tie than to incorrectly determine a winner based on minor formatting or image parsing differences. **Content accuracy of the main text is the ONLY standard.**

Figure 10: Prompt used in OCR Arena

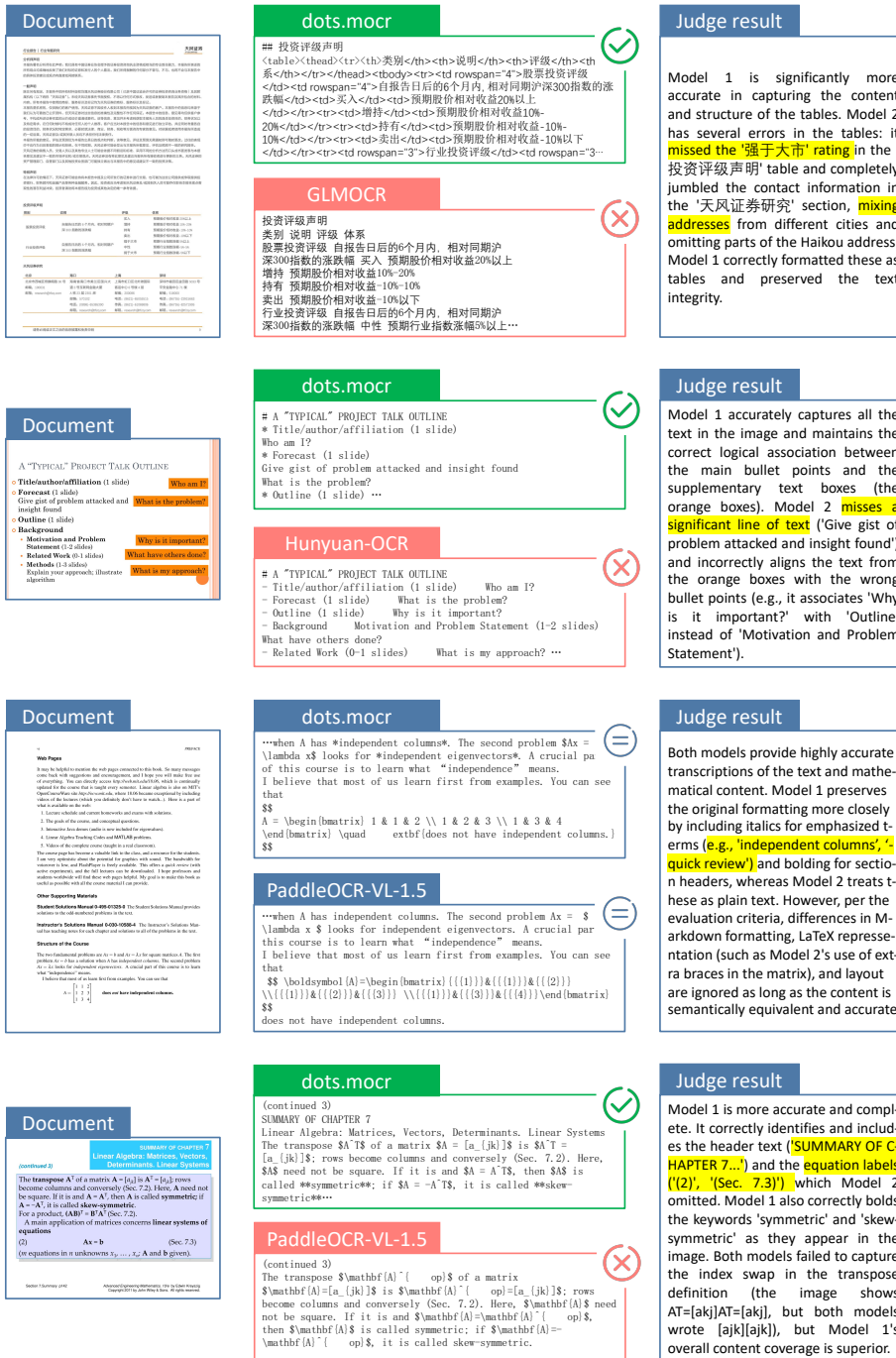


Figure 11: Example judging results in OCR Arena, showing document images, OCR outputs from two models, and the explanations produced by the VLM judge.