LaTeX2Layout: High-Fidelity, Scalable Document Layout Annotation Pipeline for Layout Detection

Feijiang Han, Zelong Wang, Bowen Wang, Xinxin Liu, Skyler Cheung, Delip Rao, Chris Callison-Burch, Lyle Ungar
University of Pennsylvania

Abstract

General-purpose Vision-Language Models (VLMs) are increasingly integral to modern AI systems for document understanding, yet their ability to perform fine-grained layout analysis remains severely underdeveloped. Overcoming this requires a large-scale, high-fidelity training dataset. However, current annotation methods, which rely on parsing rendered PDFs, are costly, error-prone, and fail to scale effectively. This work introduces a paradigm shift in data acquisition to resolve this bottleneck. We present **`LaTeX2Layout`**, a novel and generalizable procedural pipeline that obtains ground-truth layout information not from the final PDF, but directly from the LaTeX compilation process itself. By instrumenting the compiler, our method produces pixel-perfect bounding boxes and reading order, entirely bypassing the ambiguities of post-rendering parsers. This efficient and accurate pipeline enables us to generate a massive dataset of 140K pages, including 120K programmatically-generated variants that more than double the layout diversity of real-world datasets. This unique dataset allows us to fine-tune a highly efficient 3B parameter VLM, employing a curriculum learning strategy that re-ranks training examples from simple to complex layouts to optimize convergence. Our model establishes a new state-of-the-art, achieving a Kendall's Tau of 0.95 for reading order and a mAP@0.5 of 0.91 for element grounding—a nearly 200% relative improvement over formidable zero-shot baselines like GPT-4o and Claude-3.7. To foster reproducible research and future innovation, we make our data generation pipeline, dataset, and all models openly available.

1 Introduction

The widespread adoption of Large Language Models (LLMs) and retrieval-augmented generation (RAG) systems has created urgent demand for accurate PDF document understanding (Gao et al. 2023; Sharma 2025). While PDF layouts, often encoding rich semantic information, are optimized for human readability, they pose significant challenges for automated processing (Xie et al. 2024). These challenges arise from parsing complex multi-column layouts, tables, figures, mathematical formulas, and cross-references (Zhong, Tang, and Jimeno Yepes 2019; Li et al. 2020; Dolfi et al. 2022; Sun et al. 2025) while preserving semantic relationships – an essential requirement for downstream applications (Tkaczyk, Zhong et al. 2022; ComPDF 2025; Liu et al. 2024; Wang et al. 2022). Document layout models like LayoutLM (Xu et al. 2020; Huang et al. 2022) and PP-DocLayout (Sun et al. 2025) depend on large-scale annotated datasets.

However, most existing datasets derive layout annotations directly from PDFs, facing two major challenges: **(1) Reliance on expensive annotations:** Manual annotation for document layout extraction (e.g., DocLayNet (Dolfi et al. 2022), D^4LA (Da et al. 2023)) is expensive at the scale necessary for training high-quality models. **(2) Error-prone semi-automatic methods:** Semi-automatic methods that combine PDF parsers with LaTeX/XML markup (e.g., PubLayNet (Zhong, Tang, and Jimeno Yepes 2019), DocBank (Li et al. 2020), and TableBank (Li et al. 2019b)) still depend on these parsers, which frequently mislocalize elements and produce incorrect reading orders, particularly in scientific documents (Adhikari and Agarwal 2025).[1]

To address these challenges head-on, we propose **`LaTeX2Layout`**, a procedural pipeline that produces pixel-perfect layout annotations for elements in a PDF document. We achieve this by exploiting a **key insight:** *The LaTeX compiler is aware of the bounding boxes and layout information during the compilation process, and this can be used to generate ground-truth annotations without additional parsing.*

Instead of treating PDF parsing as a general computer vision problem and annotating images of PDF pages with bounding box information, like current methods, we instrument the LaTeX document to capture structural metadata during compilation. This enables our method to obtain accurate layout and reading order annotations at scale. Moreover, LaTeX's modular source code enables procedural data generation for a diverse synthetic dataset creation — We can arbitrarily vary layouts, fonts, and element arrangements while programmatically obtaining aligned annotations for data augmentation.

Our primary contributions are:

1. **A cheap, reliable, and generalizable annotation technique**: We introduce a fully procedural pipeline, **`LaTeX2Layout`**, that eliminates the need for external human annotation and PDF parsers and extracts accurate

[1] See DocBank Issues 17 and 27:
https://github.com/doc-analysis/DocBank/issues/17
https://github.com/doc-analysis/DocBank/issues/27

layout annotations at 2.5 pages per second on commodity CPU hardware. Our pipeline can generalize beyond academic papers, enabling annotations for other LaTeX-based documents such as resumes, newspapers, and textbooks.

2. **A procedural data generation framework and principled training strategy for layout detection**: We develop programmatic LaTeX perturbations to create diverse document variants with automatic annotations, resulting in a synthetic dataset that offers twice the layout diversity of real-world datasets. Furthermore, we are the first to successfully apply curriculum learning to document layout analysis by re-ranking our dataset, which we show significantly accelerates convergence and improves final model performance in fine-tuning general-purpose vision-language models (VLMs).
3. **State-of-the-art results**: We demonstrate the effectiveness of our data and methods by fine-tuning a 3B-parameter general-purpose VLM[2]. Our resulting model establishes a new state-of-the-art, achieving a Kendall's Tau of 0.95 for reading order and mAP@0.5 of 0.91 for element grounding. This represents a nearly **200% relative improvement** over zero-shot VLM baselines like GPT-4o and Claude-3.7. It also demonstrates superior robustness to visual perturbations and generalization to unseen elements compared to specialized models.
4. **Open-sourced pipeline, datasets, and models**: To foster reproducible research and spur further innovation in high-fidelity document understanding, we release our entire data generation pipeline, the 140K annotated dataset, all model weights, and the training code[3].

2 Related Work

2.1 Layout Dataset Construction via PDF Parsing

Existing layout annotation pipelines rely on external PDF parsers (PDFMiner, PDFPlumber, PyMuPDF) to extract layout metadata. PubLayNet (Zhong, Tang, and Jimeno Yepes 2019) aligns PubMed XML with PDFs, while DocBank (Li et al. 2020), DocGenome (Chen et al. 2024), and LaTeX Rainbow (Duan, Tan, and Bartsch 2023) use color-tagging in LaTeX source followed by PDF post-processing. These approaches enable scalability but inherit parser inaccuracies, particularly for complex layouts and mathematical content. Our method extracts layout metadata directly from the LaTeX compiler, eliminating parsing errors and providing deterministic annotations.

2.2 Synthetic Layout Data Generation

Prior synthetic generation methods employ deep generative models trained on real data. Early GAN-based approaches like LayoutGAN (Li et al. 2019a) and DocSynth (Biswas et al. 2021) produced low-resolution outputs with limited utility. Recent advances treat layout as sequence modeling: LayoutDM (Chai, Zhuang, and Yan 2023) uses diffusion on token sequences, while DocSynthv2 (Biswas et al. 2024) employs autoregressive transformers. Other approaches include GNNs for relational modeling (Agarwal et al. 2024), LLM-driven generation (RIDGE (Jiang et al. 2025)), and algorithmic methods like 2D bin packing (Zhao et al. 2024a). These methods require significant computational resources and struggle with complex layouts. By programmatically generating LaTeX source, we leverage a mature typesetting engine to produce structurally correct documents with perfect annotations as a deterministic compilation byproduct, eliminating expensive annotation steps.

[2]Based on Qwen-2.5-VL-3B

[3]URL withheld for blind review

2.3 Document Layout Analysis Models

Recent models jointly model textual, visual, and spatial features. The LayoutLM series (Xu et al. 2020, 2021; Huang et al. 2022) extends transformers with layout embeddings. LiLT (Wang, Jin, and Ding 2022) enables cross-lingual understanding, while LayoutLLM (Huang et al. 2023) integrates layout into LLMs. Detection-based approaches include DocLayout-YOLO (Zhao et al. 2024b) and PP-DocLayout (Du et al. 2025). Despite their success, these models require domain-specific architectures. Although VLMs have been widely applied in various document understanding systems, the potential of general-purpose VLMs for layout understanding remains largely unexplored (Zhang et al. 2024; Zong et al. 2025). Our work addresses this gap by fine-tuning a general-purpose VLM with high-fidelity layout data, demonstrating significant improvements in grounding and reading order prediction.

3 The LaTeX2Layout Pipeline

While LaTeX compilation is deterministic, standard toolchains lack a native mechanism for exporting fine-grained layout metadata, such as element bounding boxes and reading order. To address this gap, we introduce Latex2Layout, a pipeline that instruments the compilation process itself to extract high-fidelity layout annotations. Our approach treats LaTeX as a programmable typesetting engine; we inject lightweight, layout-aware commands into the source code that instruct the compiler to emit precise spatial metadata during a single compilation pass.

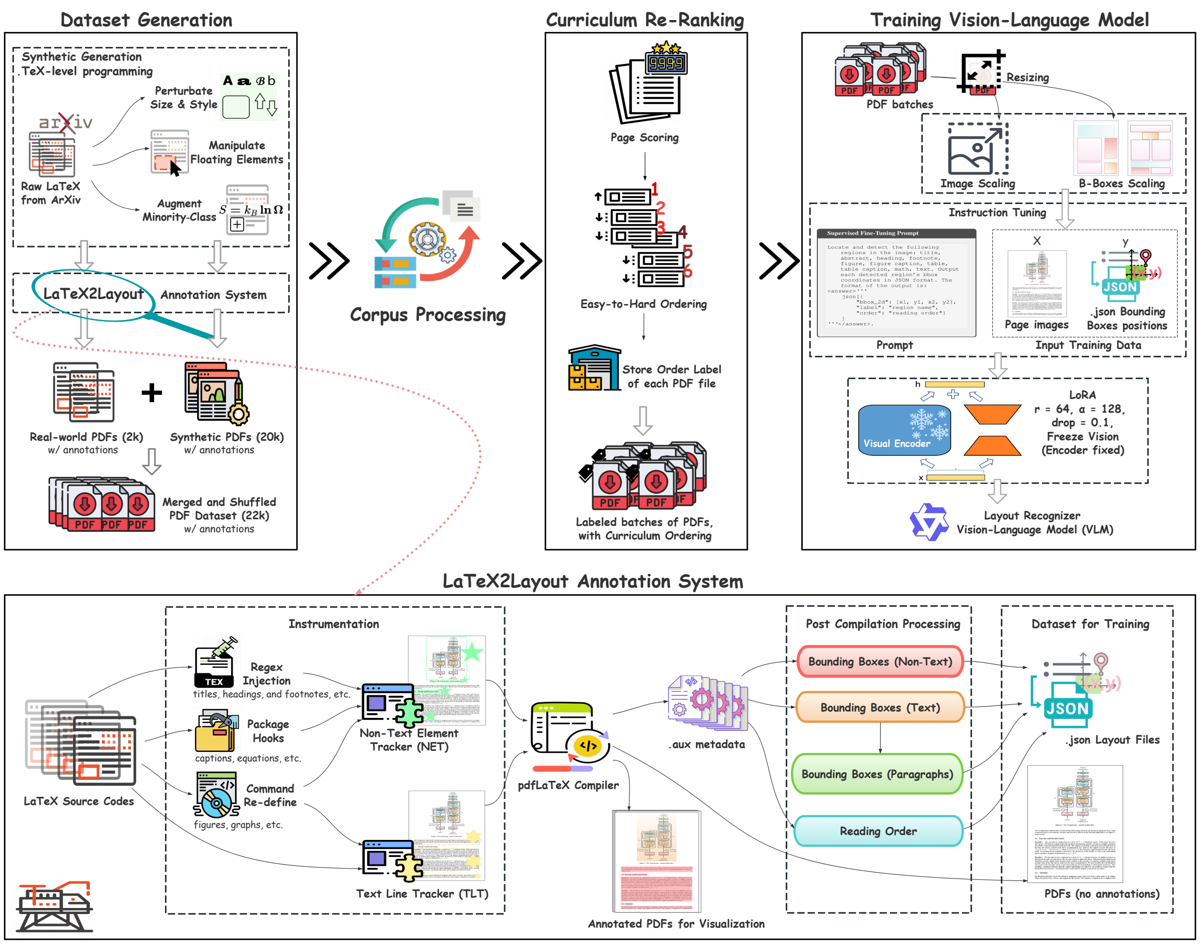
The Latex2Layout pipeline operates in two stages:

- **LaTeX-Level Instrumentation:** Two custom packages – the Non-Text Element Tracker (NET) and the Text Line Tracker (TLT) – annotate source content to record layout metadata as part of the compilation process.
- **Post-Compilation Processing:** The emitted metadata is parsed to compute bounding boxes, align coordinate systems, and reconstruct reading order.

3.1 Tracking Non-Text Elements

The NET extracts layout metadata for non-paragraph elements in LaTeX documents. These include titles, headings, figures, tables, captions, and mathematical expressions. Such elements typically follow consistent syntactic patterns, making them well-suited for systematic instrumentation. Implementation details are provided in Appendix A.1.

Inspired by the LaTeX \fbox command that visually frames content, we introduce a novel command \layoutmark that


Dataset Generation
Synthetic Generation
.TeX-level programming
Perturbate Size & Style
Manipulate Floating Elements
Augment Minority-Class
Raw LaTeX from ArXiv
LaTeX2Layout
Annotation System
Real-world PDFs (2k) w/ annotations
Synthetic PDFs (20k) w/ annotations
Merged and Shuffled PDF Dataset (22k) w/ annotations
Corpus Processing
Curriculum Re-Ranking
Page Scoring
Easy-to-Hard Ordering
Store Order Label of each PDF file
Labeled batches of PDFs, with Curriculum Ordering
Training Vision-Language Model
PDF batches
Resizing
Image Scaling
B-Boxes Scaling
Instruction Tuning
Prompt
Page images
.json Bounding Boxes positions
Input Training Data
LoRA
r = 64, α = 128, drop = 0.1, Freeze Vision (Encoder fixed)
Visual Encoder
Layout Recognizer Vision-Language Model (VLM)
LaTeX2Layout Annotation System
Instrumentation
Regex Injection
titles, headings, and footnotes, etc.
Package Hooks
captions, equations, etc.
Command Re-define
figures, graphs, etc.
LaTeX Source Codes
Non-Text Element Tracker (NET)
Text Line Tracker (TLT)
pdfLaTeX Compiler
.aux metadata
Annotated PDFs for Visualization
Post Compilation Processing
Bounding Boxes (Non-Text)
Bounding Boxes (Text)
Bounding Boxes (Paragraphs)
Reading Order
Dataset for Training
.json Layout Files
PDFs (no annotations)

Supervised Fine-Tuning Prompt

~~~
Locate and detect the following
    regions in the image: title,
    abstract, heading, footnote,
    figure, figure caption, table,
    table caption, math, text. Output
    each detected region's bbox
    coordinates in JSON format. The
    format of the output is:
<answer>```
     json[{
          "bbox_2d": [x1, y1, x2, y2],
          "label": "region name",
          "order": "reading order"}
     ]
```</answer>.
~~~

Figure 1: An overview of our pipeline for dataset generation and model training. The figure is split into the main workflow (top) and our core annotation system (bottom). **(1) Dataset Generation**: We crawl and synthetically perturb LaTeX source files from arXiv. Our **`LaTeX2Layout` Annotation System** (magnified below) compiles these sources, instrumenting them to extract pixel-perfect layout annotations directly from the compiler. **(2) Curriculum Re-Ranking**: The resulting dataset is scored by layout complexity and sorted into an easy-to-hard curriculum. **(3) VLM Training**: Finally, a VLM is instruction-tuned on the ordered curriculum to predict bounding boxes and reading order from page images.

silently records the layout of the wrapped element during compilation. It captures the following metadata: (a) Baseline Coordinates; (b) Box Dimensions, including width, height, and depth. [4]. All metadata is written to the auxiliary *.aux* file [5], enabling structured post-processing. Appendix A.4 provides the specific format of this compiler output.

Instrumentation Methods The NET is compatible with any context where the \fbox command can be applied. We support three flexible instrumentation strategies to accommodate the diverse range of elements. Full implementation examples are available in Appendix A.2.

- **Regex-Based Injection:** For elements with simple and consistent syntax, such as titles, abstracts, and headings, we use regular expressions to automatically find and wrap them with our \layoutmark command. However, this external matching approach can be brittle for elements with more complex or variable syntax. For these cases, we employ two more robust methods.
- **Package Hook Integration:** Many LaTeX packages provide internal hooks or options that we leverage for more reliable instrumentation. For example, the *caption* package allows us to define a custom format that automatically applies \layoutmark to every figure and table caption. Similarly, for complex multi-line mathematical content, the *empheq* package's *box* option can be hooked to guarantee a precise formula box.
- **Command Redefinition:** As a final and universally applicable strategy, we use LaTeX's native command redefinition mechanism. This approach is ideal for core commands like \includegraphics that are too syntactically diverse for regex and lack convenient package hooks. We redefine the target command to first wrap its content with \layoutmark and then execute its original functionality.

[4]In LaTeX, height refers to the portion above the baseline, and depth refers to the portion below.

[5]The `.aux` file is a compiler-generated side file typically used for cross-referencing. We extend it to store layout information.

3.2 Tracking Text Lines

While NET works well for structured non-text elements, plain text paragraphs pose a different challenge. They lack explicit LaTeX markup, making them difficult to identify using pattern matching. To address this, we introduce the TLT. Inspired by the *lineno* package, which numbers lines in PDFs after compilation, TLT redefines its internal hook *MakeLineNo* to track line-level positional metadata automatically. This process requires no changes to the paragraph structure and no manual commands. Implementation details are provided in Appendix A.3.

Each text line is annotated with the following metadata: (a) Baseline Coordinates: the position of the line's lower-left corner; (b) Paragraph ID: a unique identifier for the LaTeX paragraph the line belongs to; (c) Column Number: the column index of the line in the rendered PDF, useful for detecting column breaks; (d) Line Height and Width: the vertical and horizontal extent of the rendered text. The exact format for this line-level metadata is also detailed in Appendix A.4.

3.3 Layout Extraction

The layout metadata recorded in the auxiliary (*.aux*) file uses the LaTeX coordinate system. To obtain an accurate PDF-level layout, we perform coordinate unit conversion and axis transformation. Additionally, text lines are aggregated into paragraph-level segments, with support for handling column and page breaks.

Bounding Box Computation For elements tracked by the NET, the bounding box is computed as:

$$\mathbf{B}_{\text{nontext}} = \begin{bmatrix} x_{\text{base}} \\ y_{\text{base}} + B_h \\ x_{\text{base}} + B_w \\ y_{\text{base}} - B_d \end{bmatrix} \tag{1}$$

Here, B_h is the height above the baseline, B_d is the depth below the baseline, and B_w is the element width.

For individual text lines tracked by TLT, the bounding box is computed as:

$$\mathbf{B}_{\text{text}} = \begin{bmatrix} x_{\text{line}} \\ y_{\text{line}} + L_h \\ x_{\text{line}} + L_w \\ y_{\text{line}} \end{bmatrix} \tag{2}$$

Here, L_h is the height of the rendered line, and L_w is its width.

Paragraph-Level Aggregation Paragraphs are reconstructed by grouping text lines with the same paragraph ID. To capture higher-level semantics, adjacent paragraphs can be merged if their vertical spacing falls below a configurable threshold.

Cross-Column and Cross-Page Linking The TLT module tracks the column and page number of each line, enabling accurate reconstruction of paragraph structures that span across columns or pages. This allows the system to distinguish true paragraph breaks from layout-induced ones.

Coordinate Transformation LaTeX reports positions in scaled points (sp), while PDF uses pixels. We convert coordinates using the following equations:

$$\begin{aligned} \text{Points (pt)} &= \frac{\text{Scaled Points (sp)}}{65536}, \\ \text{Inches} &= \frac{\text{Points (pt)}}{72.27}, \\ \text{Pixels} &= \text{Inches} \times \text{DPI} \end{aligned} \tag{3}$$

Here, DPI (dots per inch) specifies the rendering resolution and defines the pixel-to-physical length ratio.

Additionally, the LaTeX coordinate system uses a bottom-left origin, whereas the PDF coordinate system uses a top-left origin. Therefore, the y-coordinate is adjusted as:

$$y_{\text{PDF}} = \text{Page Height} - y_{\text{LaTeX}} \tag{4}$$

Reading Order Alignment Unlike heuristic PDF parsers that assume fixed reading flows (e.g., top-to-bottom or left-to-right), we derive reading order directly from the compilation sequence. This approach preserves authorial intent and resolves ambiguous layout cases. For example, when a paragraph references a footnote, LaTeX compiles the referencing text line first, immediately followed by the footnote layout, even if the footnote appears at the bottom of the page.

4 Dataset Generation

4.1 Constructing the Real-World Seed Dataset

To build a foundational dataset of contemporary scientific documents, we curated a 20K-page seed dataset by crawling LaTeX sources from arXiv papers published between 2022 and 2024. To ensure a representative distribution of layouts, we balanced the corpus between single-column (e.g., NeurIPS, ICML) and double-column (e.g., AAAI, ACL) formats from premier venues. We then processed each compilable source through our Latex2Layout pipeline to extract high-fidelity layout and reading order annotations. Appendix B provides a detailed breakdown of this dataset.

4.2 Synthetic Augmentation

We identified two primary challenges with relying solely on crawled data: the collection process (crawling, cleaning, and validation) is laborious, and the resulting dataset typically exhibits an imbalanced, long-tail distribution of element types that is difficult to control. To address this, we introduce a procedural augmentation method that efficiently generates a more complex, diverse, and balanced dataset. Figure 2 showcases examples of documents generated using this method, along with their perfectly aligned annotations.

This approach leverages the core reusability of our trackers: once a LaTeX source file is instrumented, generating new, perfectly annotated layout variants is as simple as modifying styles or reordering elements and recompiling. This bypasses the need to re-run the entire instrumentation pipeline for each variant, enabling massive-scale data generation.

Using this method, we generated a 120K-page augmented dataset. As detailed in Appendix B, this new corpus more

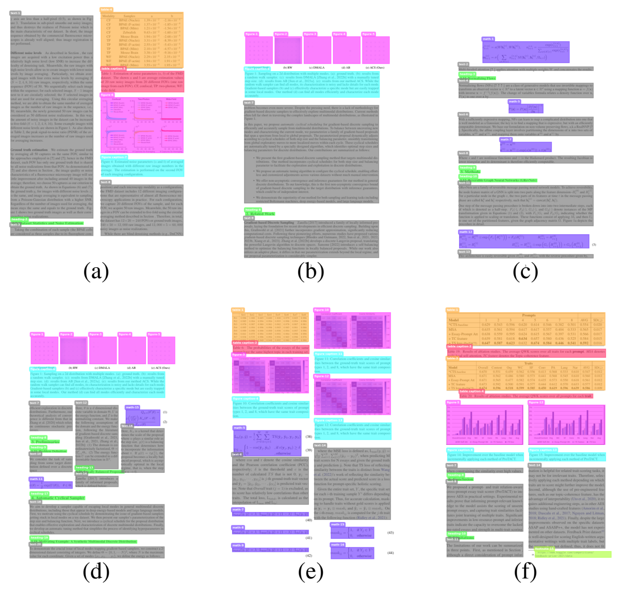


Figure 2: High-quality layout annotations from our LaTeX2Layout pipeline. Top row: Examples from crawled real-world datasets. Bottom row: Examples from procedurally generated synthetic datasets. Additional synthetic examples are in Appendix G, with generalizations to broader document types in Appendix H.

than doubles the number of unique page-level element combinations and nearly doubles the aggregate proportion of critical minority classes. We employ three strategies:

- **Size and Style Perturbation (30K):** We apply controlled visual perturbations by scaling floating elements (figures, tables, equations) and adjusting page-level styles (e.g., margins, fonts). This enriches layout diversity while preserving the original semantic structure, creating novel layouts that are often more challenging for the model to parse than their original counterparts.
- **Floating Element Manipulation (80K):** Floating elements create rich layouts by displacing the natural flow of text. We amplify this effect by randomly reordering and duplicating floats within documents. These manipulations generate complex and varied float–text interactions, exposing the model to a wider range of challenging spatial arrangements.
- **Minority Element Augmentation (10K):** To directly counteract class imbalance, we generate pages composed exclusively of underrepresented elements (e.g., figures, tables, footnotes, formulas). This provides focused, targeted supervision for these critical minority categories, preventing them from being overlooked during training.

4.3 Curriculum Learning Re-Ranking

Document layouts possess an intrinsic and measurable difficulty gradient, ranging from simple, single-column pages to dense layouts with complex, interwoven elements. We argue that this inherent difficulty gradient is ideally suited for leveraging the power of curriculum learning (CL)—a training paradigm that guides a model to learn progressively from simple to more complex examples to optimize final convergence and performance. While CL has proven effective in other domains (Bengio et al. 2009; Saha et al. 2024), our work extends this strategy to the task of document layout parsing.

Specifically, we re-rank our entire training dataset (both real and synthetic) by scoring page complexity based on the number and heterogeneity of its constituent elements. This graduated ordering, from simple to complex, allows the model to first master a foundational visual grammar—such as basic text blocks and headings—before progressing to more challenging structures like multi-column floats and dense tables. Empirically, we find that this approach not only accelerates convergence but also improves the final model's performance and training stability.

5 Model Training and Evaluation

Training. We adopt Qwen-2.5-VL-3B as the backbone of our vision-language system, given its strong performance and growing adoption in recent multimodal research (Bai et al. 2025). Our core training strategy involves freezing the pre-trained vision encoder and using Low-Rank Adaptation (LoRA) (hiyouga 2023) to efficiently fine-tune only the language model. This approach focuses the training on aligning the model's powerful visual features with our desired structured JSON output, which specifies each element's bounding box, semantic label, and reading order. The model was trained on our combined dataset of 140K document images. Further details on image preprocessing, prompt design, and specific hyperparameters are available in Appendix C.1.

Evaluation. Following previous works (Zhao et al. 2024b; Chen et al. 2024), our evaluation assesses two distinct capabilities: element grounding and reading order prediction. We use Average Precision (AP) with a 0.5 IoU threshold to measure grounding accuracy and Kendall's Tau (τ) (Lapata 2006) [6] to evaluate the correctness of the predicted reading order. A key step in our pipeline is adapting the generative VLM for standard evaluation. Since the model does not produce explicit confidence scores, we propose a method to derive them from the token probability of the predicted class. These scores are then used with class-wise Non-Maximum Suppression (NMS) to refine detections before computing the final metrics. The full evaluation protocol is detailed in Appendix C.2.

6 Experimental Analysis

6.1 Evaluation Setup

Dataset. To ensure a realistic evaluation, we curated a test set of 1K real-world academic document pages following the same data standard as the training set. Ground-truth annotations were derived using our LaTeX2Layout pipeline and

[6]Kendall's Tau (τ) is a rank correlation coefficient that measures the ordinal association between two sequences, ranging from -1 (perfect disagreement) to 1 (perfect agreement). This helps evaluate how well the predicted reading order matches the ground truth.

were manually verified to ensure accuracy and consistency. Detailed statistics are available in Appendix B.

Models. We evaluate (1) zero-shot performance of general-purpos VLMs (GPT-4o (OpenAI 2024), Claude-3.7-Sonnet (Anthropic 2024), and Qwen-2.5VL-3B-Instruct (Qwen-2.5VL) (Bai et al. 2025)), (2) Qwen-2.5VL fine-tuned on 20K real and 140K (20K real + 120K synthetic) pages, and (3) YOLOv8 trained on the same real and synthetic datasets for grounding-only comparison.

6.2 Main Results

Our experiments are designed to validate the annotations from our Latex2Layout pipeline and to demonstrate the impact of our synthetic data on model performance. We evaluate across 11 element categories, with full results in Table 1.

First, we observe that general-purpose VLMs struggle with specialized layout parsing in a zero-shot setting. Leading models like GPT-4o and Claude-3.7 achieve low τ scores (0.31 and 0.32, respectively) and mAP50 scores (0.34 and 0.29, respectively), especially on fine-grained categories like captions and footnotes. The smaller Qwen-2.5VL-3B performed even more poorly; its limited pre-training for structured output tasks often resulted in it failing to adhere to the prompt, generating incomplete or improperly formatted output. This underscores that without task-specific fine-tuning, general VLMs lack the necessary spatial inductive biases for this task.

Our results demonstrate that fine-tuning VLMs with layout-specific data derived from our Latex2Layout pipeline can effectively improve their performance on this task. Fine-tuning Qwen-2.5VL on our 20K real-world samples achieves a reading order τ of 0.91 and an element grounding mAP50 of 0.78. However, performance on rare or small categories like footnote and math remains limited, a result we attribute to the inherent class imbalance and limited layout diversity in the crawled real-world data.

Augmenting the training set with our 120K synthetic samples effectively addresses this long-tail problem and pushes the model to a new level of performance, achieving a final τ of **0.95** and an mAP50 of **0.91**. This combined dataset yields dramatic relative gains on previously challenging categories, including a 53% improvement for footnote and 64% for math. These results confirm that the rich and complex layout diversity introduced by our synthetic data is critical for robust document understanding.

6.3 Ablation Studies

We conduct two ablation studies to isolate and analyze the contributions of our core components: the use of synthetic data and our curriculum learning strategy.

Effectiveness of Synthetic Data: A Scaling Analysis To rigorously evaluate the value of our synthetic data, we compare models trained exclusively on synthetic datasets of increasing sizes against a baseline trained on 20K real-world samples. The results, shown in Figure 3, reveal two key insights.

First, performance on reading order prediction saturates quickly. A model trained on just 20K synthetic samples

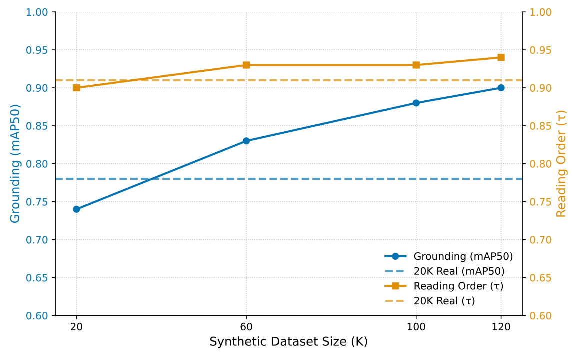


Figure 3: Impact of Synthetic Dataset Size on Grounding (mAP50) and Reading Order (τ) Compared to a 20K Real Data Baseline

achieves a Kendall's Tau (τ) of 0.90, nearly matching the baseline's 0.91. This suggests the underlying structural logic of academic layouts is highly consistent and can be learned from a moderately sized dataset.

Second, for the more challenging task of element grounding, our results demonstrate a clear and promising scaling effect: performance improves consistently as the volume of synthetic data increases. As shown in Figure 3, we observe an interesting phenomenon at the 20K data point: the model trained on synthetic data slightly lags behind the model trained on real data. We attribute this to the fact that with smaller dataset sizes, the advantage of the real data's closer **distributional alignment** to the test set outweighs the initial benefits of the synthetic data's **diversity**.

However, our results convincingly show that as the synthetic dataset scales, the immense value of structural diversity becomes the dominant factor. This diversity-driven improvement progressively closes the initial gap and ultimately propels the model to a superior final performance, significantly outperforming the real-data baseline. **This finding offers a key practical takeaway:** *substantial performance gains are driven not by the visual novelty of individual elements, but by the richness and complexity of the layouts they inhabit. It validates the strategy of programmatically generating vast and diverse training data by simply composing existing elements into novel structural arrangements.*

Effectiveness of Curriculum Learning To isolate the impact of our curriculum learning (CL) strategy, we compare training on our 20K real-world dataset in an easy-to-hard order versus a standard random order. The results confirm that our CL strategy enhances both training efficiency and final model performance.

First, the CL approach improves training dynamics. As shown in Figure D.4 in Appendix D.5, the model trained with our curriculum exhibits significantly faster convergence and achieves a lower final training loss, indicating a more stable and effective learning process.

Second, as shown in Table 2, this improved training stability translates directly to superior downstream accuracy, increasing reading order τ from 0.88 to 0.91 and boosting the average element grounding mAP50 from 0.72 to 0.78.

Table 1: Main results for Reading Order Prediction (τ) and Element Grounding (mAP50). Our fine-tuned models, trained with data from Latex2Layout, establish a new state-of-the-art, dramatically outperforming leading general-purpose VLMs.

Model	Reading Order (τ)	Element Grounding (mAP50)											
		Tit	Abs	Head	Txt	Math	Fig	Tab	FigCap	TabCap	Foot	Ref	Average
Zero-Shot Baselines													
GPT-4o	0.31	0.58	0.37	0.08	0.56	0.18	0.57	0.35	0.17	0.21	0.09	0.62	0.34
Claude-3.7	0.32	0.65	0.58	0.05	0.48	0.08	0.38	0.39	0.01	0.07	0.12	0.41	0.29
Qwen-2.5VL (Base)	0.12	0.09	0.00	0.07	0.21	0.17	0.09	0.03	0.03	0.00	0.06	0.00	0.07
Ours (Qwen-2.5VL Fine-tuned)													
w/ 20K Real Samples	0.91	0.75	0.85	0.73	0.92	0.55	0.82	0.85	0.84	0.78	0.57	0.92	0.78
+ 120K Synthetic Samples	**0.95**	**0.86**	**0.95**	**0.83**	**0.95**	**0.90**	**0.93**	**0.92**	**0.93**	**0.92**	**0.87**	**0.97**	**0.91**

Table 2: Ablation study on the effectiveness of Curriculum Learning, conducted on the 20K real-world dataset. Using CL improves both reading order prediction and overall element grounding accuracy. Best scores for each metric are shown in **bold**.

Training Strategy	Reading Order (τ)	Element Grounding (AP50)											
		Tit	Abs	Head	Txt	Math	Fig	Tab	FigCap	TabCap	Foot	Ref	Average
Standard Training (Random)	0.88	**0.77**	0.78	0.66	0.87	0.44	0.78	0.83	0.84	**0.81**	0.28	0.85	0.72
Curriculum Learning (Re-Ranking)	**0.91**	0.75	**0.85**	**0.73**	**0.92**	**0.55**	**0.82**	**0.85**	0.84	0.78	**0.57**	**0.92**	**0.78**

The benefit is most striking for challenging, low-frequency categories; for instance, the mAP50 for footnote more than doubles from 0.28 to 0.57.

We hypothesize this substantial gain stems from a more efficient, compositional learning process induced by the curriculum. In a standard random regimen, the model may simultaneously learn to identify all element types while parsing their complex spatial relationships on a difficult page, which may cause losses from minority categories to be overwhelmed by dominant ones. Our curriculum decouples these tasks. The model first learns to robustly identify foundational elements in simple layouts where the learning signal is clear. Then, when presented with more complex pages, it can leverage this existing knowledge and focus its capacity on parsing more intricate arrangements or recognizing new element types, rather than re-learning everything from scratch. These results validate that CL is a simple yet highly effective method for improving document layout model training.

7 Further Analysis and Discussion

To provide a clearer picture of our framework's performance and potential, we conducted an in-depth analysis covering three key areas: a comparative evaluation of our VLM against specialized computer vision detectors, a detailed error analysis of the VLM, and an efficiency analysis of our model and Latex2Layout pipeline. We summarize the main findings below, with full technical details available in Appendix D.

1. **VLM Performance and Versatility.** Our VLM not only achieves grounding accuracy competitive with the specialized YOLOv8 detector but also offers significant advantages in versatility. It uniquely performs multi-task learning (simultaneously grounding elements and predicting reading order), shows greater robustness to visual noise and out-of-distribution (OOD) layouts, and demonstrates capacity for zero-shot generalization to new categories (see Appendix D.1 for details).
2. **Limitations and Future Directions.** Our error analysis identifies specific limitations that highlight clear paths for future improvement. These include the omission of very small objects (addressable via targeted data synthesis), inaccuracies for OOD layouts (fixable by expanding the training corpus using our Latex2Layout's generalization capabilities), and sensitivity to severe visual noise (which can be improved with data augmentation). We also pinpoint a nuanced training challenge where the model's confidence doesn't always align with its spatial accuracy, pointing to the need for spatially-aware training objectives (see Appendix D.2 and D.3).
3. **Pipeline Efficiency.** From a practical standpoint, our data extraction pipeline is highly efficient, processing each page in just **0.435 seconds** on a standard consumer CPU. This confirms its suitability for creating large-scale datasets affordably and at scale (see Appendix D.4).

8 Conclusion

This work introduces LaTeX2Layout, a novel and generalizable pipeline for extracting high-fidelity layout annotations directly from the LaTeX compilation process, eliminating the need for post-rendering PDF parsing. Our method offers significant improvements in annotation accuracy and scalability over traditional approaches. By enabling the generation of diverse synthetic datasets through LaTeX's flexible source code modifications, we pave the way for training robust models capable of understanding complex document layouts. Future work will explore further optimization of layout perturbation strategies and the extension of our system to a broader range of document types and formats.

References

Adhikari, N. S.; and Agarwal, S. 2025. A Comparative Study of PDF Parsing Tools Across Diverse Document Categories. arXiv:2410.09871.

Agarwal, A.; Patel, H.; Pattnayak, P.; Panda, S.; Kumar, B.; and Kumar, T. 2024. Enhancing document ai data generation through graph-based synthetic layouts. *arXiv preprint arXiv:2412.03590*.

Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku. https://www.anthropic.com/research/claude-3-model-family.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; Zhong, H.; Zhu, Y.; Yang, M.; Li, Z.; Wan, J.; Wang, P.; Ding, W.; Fu, Z.; Xu, Y.; Ye, J.; Zhang, X.; Xie, T.; Cheng, Z.; Zhang, H.; Yang, Z.; Xu, H.; and Lin, J. 2025. Qwen2.5-VL Technical Report. *arXiv preprint arXiv:2502.13923*.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48.

Biswas, S.; Jain, R.; Morariu, V. I.; Gu, J.; Mathur, P.; Wigington, C.; Sun, T.; and Lladós, J. 2024. Docsynthv2: A practical autoregressive modeling for document generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8148–8153.

Biswas, S.; Riba, P.; Lladós, J.; and Pal, U. 2021. Docsynth: a layout guided approach for controllable document image synthesis. In *International Conference on Document Analysis and Recognition*, 555–568. Springer.

Chai, S.; Zhuang, L.; and Yan, F. 2023. Layoutdm: Transformer-based diffusion model for layout generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18349–18358.

Chen, Z.; Wang, S.; Wang, J.; et al. 2024. DocGenome: An Open Large-scale Scientific Document Benchmark for Training and Testing Multi-modal Large Language Models. *arXiv preprint arXiv:2406.11633*.

ComPDF. 2025. What is PDF Document Layout Analysis (DLA)? Available at https://www.compdf.com/blog/pdf-document-layout-analysis.

Da, C.; Luo, C.; Zheng, Q.; and Yao, C. 2023. Vision Grid Transformer for Document Layout Analysis. In *ICCV*.

Dolfi, M.; et al. 2022. DocLayNet: A Large Human-Annotated Dataset for Document-Layout Analysis. *arXiv preprint arXiv:2206.01062*.

Du, X.; Wang, B.; Zhao, Z.; et al. 2025. PP-DocLayout: A Unified Document Layout Detection Model with Global-to-Local Perception. *arXiv preprint arXiv:2503.17213*.

Duan, C.; Tan, Z.; and Bartsch, S. 2023. LaTeX Rainbow: Universal LaTeX to PDF Document Semantic & Layout Annotation Framework. In *Proceedings of the Second Workshop on Information Extraction from Scientific Publications*, 56–67. Bali, Indonesia: Association for Computational Linguistics.

Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H.; and Wang, H. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1).

hiyouga. 2023. LLaMA-Factory: Unified Efficient Fine-Tuning of 100+ LLMs. https://github.com/hiyouga/LLaMA-Factory.

Huang, S.; Xu, Y.; Cui, L.; et al. 2023. LayoutLLM: Layout-Aware Large Language Models for Document Understanding. *arXiv preprint arXiv:2309.00909*.

Huang, Y.; Xu, Y.; Li, L.; Cui, Y.; Zhang, J.; and Wei, F. 2022. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking.

Jiang, Z.-H.; Lin, C.-W.; Li, W.-H.; Liu, H.-T.; Yeh, Y.-R.; and Chen, C.-S. 2025. Relation-Rich Visual Document Generator for Visual Information Extraction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 14449–14459.

Lapata, M. 2006. Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4): 471–484.

Li, J.; Yang, J.; Hertzmann, A.; Zhang, J.; and Xu, T. 2019a. Layoutgan: Generating graphic layouts with wireframe discriminators. *arXiv preprint arXiv:1901.06767*.

Li, M.; Cui, L.; Huang, S.; Wei, F.; Zhou, M.; and Li, Z. 2019b. TableBank: A Benchmark Dataset for Table Detection and Recognition. *arXiv preprint arXiv:1903.01949*.

Li, X.; Xu, Y.; Cui, L.; Huang, S.; Wei, F.; Zhou, M.; Li, M.; and Wang, D. 2020. DocBank: A Benchmark Dataset for Document Layout Analysis. *arXiv preprint arXiv:2006.01038*.

Liu, Y.; et al. 2024. AutoIE: An Automated Framework for Information Extraction from Scientific Literature. *arXiv preprint arXiv:2401.16672*.

OpenAI. 2024. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/. Accessed: 2025-05-26.

Qwen Team. 2025a. LLaMA-Factory - Qwen docs. https://qwen.readthedocs.io/en/latest/training/llama_factory.html.

Qwen Team. 2025b. Qwen/Qwen2.5-VL-3B-Instruct. https://huggingface.co/Qwen/Qwen2.5-VL-3B-Instruct.

Saha, R.; Fahim, A.; Fyshe, A.; and Murphy, A. 2024. Exploring Curriculum Learning for Vision-Language Tasks: A Study on Small-Scale Multimodal Training. *arXiv preprint arXiv:2410.15509*.

Sharma, C. 2025. Retrieval-Augmented Generation: A Comprehensive Survey of Architectures, Enhancements, and Robustness Frontiers. *arXiv preprint arXiv:2506.00054*.

Sun, T.; et al. 2025. PP-DocLayout: A Unified Document Layout Detection Model to Accelerate Large-Scale Data Construction. *arXiv preprint arXiv:2503.17213*.

Tkaczyk, D.; Zhong, X.; et al. 2022. VILA: Improving Structured Content Extraction from Scientific PDFs via Visual Layout Grouping. *Transactions of the Association for Computational Linguistics*. Available at https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00466/110438/VILA-Improving-Structured-Content-Extraction-from.

Wang, J.; Jin, L.; and Ding, K. 2022. LiLT: A Simple yet Effective Language-Independent Layout Transformer for Structured Document Understanding. *arXiv preprint arXiv:2202.13669*.

Wang, Z.; et al. 2022. A Hybrid Approach for Document Layout Analysis in Document Images. *arXiv preprint arXiv:2404.17888*.

Xie, X.; Yan, H.; Yin, L.; Liu, Y.; Ding, J.; Liao, M.; Liu, Y.; Chen, W.; and Bai, X. 2024. WuKong: A Large Multimodal Model for Efficient Long PDF Reading with End-to-End Sparse Sampling. *arXiv preprint arXiv:2410.05970*.

Xu, Y.; Li, M.; Cui, L.; Huang, S.; Wei, F.; and Zhou, M. 2020. LayoutLM: Pre-training of Text and Layout for Document Image Understanding. *KDD*.

Xu, Y.; Xu, T.; Lv, Y.; Cui, L.; Lu, Y.; Wei, F.; and Zhou, M. 2021. LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding. *arXiv preprint arXiv:2012.14740*.

Zhang, J.; Huang, J.; Jin, S.; and Lu, S. 2024. Visual Prompting in Multimodal Large Language Models: A Survey. *arXiv preprint arXiv:2409.15310*.

Zhao, Z.; Kang, H.; Wang, B.; and He, C. 2024a. Doclayout-yolo: Enhancing document layout analysis through diverse synthetic data and global-to-local adaptive perception. *arXiv preprint arXiv:2410.12628*.

Zhao, Z.; Kang, H.; Wang, B.; and He, C. 2024b. DocLayout-YOLO: Enhancing Document Layout Analysis through Diverse Synthetic Data and Global-to-Local Adaptive Perception. *arXiv preprint arXiv:2410.12628*.

Zhong, X.; Tang, J.; and Jimeno Yepes, A. 2019. PubLayNet: Largest Dataset Ever for Document Layout Analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 1015–1022. IEEE.

Zong, Y.; Zhang, Q.; An, D.; Li, Z.; Xu, X.; Xu, L.; Tu, Z.; Xing, Y.; and Dabeer, O. 2025. Ground-V: Teaching VLMs to Ground Complex Instructions in Pixels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Reproducibility Checklist

This paper:

Includes a conceptual outline and/or pseudocode description of AI methods introduced (Yes)

Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (Yes)

Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper (Yes)

Does this paper make theoretical contributions? (No)

If yes, please complete the list below.

All assumptions and restrictions are stated clearly and formally. (NA)

All novel claims are stated formally (e.g., in theorem statements). (NA)

Proofs of all novel claims are included. (NA)

Proof sketches or intuitions are given for complex and/or novel results. (NA)

Appropriate citations to theoretical tools used are given. (NA)

All theoretical claims are demonstrated empirically to hold. (NA)

All experimental code used to eliminate or disprove claims is included. (NA)

Does this paper rely on one or more datasets? (Yes)

If yes, please complete the list below.

A motivation is given for why the experiments are conducted on the selected datasets (Yes)

All novel datasets introduced in this paper are included in a data appendix. (Yes)

All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (Yes)

All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. (Yes)

All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. (Yes)

All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. (NA)

Does this paper include computational experiments? (Yes)

If yes, please complete the list below.

This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. (Partial)

Any code required for pre-processing data is included in the appendix. (Yes).

All source code required for conducting and analyzing the experiments is included in a code appendix. (Yes)

All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (Yes)

All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (Yes)

If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. (NA)

This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. (Yes)

This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. (Yes)

This paper states the number of algorithm runs used to compute each reported result. (Yes)

Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. (Yes)

The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). (No)

This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. (Yes)

A Implementation Details of Latex2Layout Trackers

A.1 Non-Text Element Tracker (NET)

The NET is responsible for extracting precise bounding box information for discrete, non-paragraph elements such as titles, figures, tables, and equations.

The core principle of NET is to leverage the LaTeX internal box model. Inspired by the native `\fbox` command, we define a new command, `\layoutmark`, that programmatically wraps target content in a measurement box. During compilation, this wrapper uses TeX primitives (`\wd`, `\ht`, `\dp`) to record the element's width, height, and depth, and the ‘zref‘ package's `\zsavepos` command to record its baseline coordinates. This metadata is then written to the auxiliary (‘.aux‘) file for post-processing, without producing any visual artifacts in the final PDF. The core implementation is shown below.

Non-Text Element Tracker

```
\makeatletter
% Define a new counter for tracking fbox instances.
\newcounter{fboxcnt}

% Backup the original \fbox command.
\let\orig@fbox\fbox

% Redefine \fbox to track dimensions of its content.
\renewcommand{\fbox}[1]{%
  % Increment the fbox counter for each instance.
  \refstepcounter{fboxcnt}

  % Measure the content inside \fbox and store it in a temporary box.
  \setbox\@tempboxa=\hbox{#1}%

  % If the box width exceeds the available width, wrap the content into a vbox.
  \ifdim\wd\@tempboxa<\dimexpr\linewidth+0.5\marginparsep+0.5\marginparwidth\relax
  \else
    \setbox\@tempboxa=\hbox{\vbox{#1}}%
  \fi

  % Write the width, height, and depth metadata to the auxiliary file.
  \protected@write\@auxout{}{\string\newlabel{width}{{\the\wd\@tempboxa}}}%
  \protected@write\@auxout{}{\string\newlabel{height}{{\the\ht\@tempboxa}}}%
  \protected@write\@auxout{}{\string\newlabel{depth}{{\the\dp\@tempboxa}}}%

  % Temporarily disable fbox padding and border thickness.
  \setlength{\fboxrule}{0pt}\setlength{\fboxsep}{0pt}%

  % Output the box, tracking the baseline position.
  \ifdim\wd\@tempboxa<\dimexpr\linewidth+0.5\marginparsep+0.5\marginparwidth\relax
     % Save baseline position for smaller boxes.
    \orig@fbox{\zsavepos{base}#1}%
  \else
     % Save baseline position for wrapped content.
    \orig@fbox{\zsavepos{base}\vbox{#1}}%
  \fi
}

% Auto-wrapping function for float content
\newcommand{\layoutmark}[1]{\fbox{#1}}%
}
\makeatother
```

A.2 NET Instrumentation Strategies

To apply `\layoutmark` systematically across a LaTeX document, we employ three complementary instrumentation strategies that accommodate a wide range of LaTeX structures.

1. Direct Injection. For elements with consistent source patterns like titles or section headings, `\layoutmark` can be inserted directly into the source code, a process that can be automated with simple regular expressions.

Heading Injection

```
\section*{
\layoutmark{
    Introduction
}}
```

2. Package Hook Integration. For elements managed by sophisticated packages, we can leverage their built-in functionality for cleaner and more robust instrumentation, often for either automation or improved accuracy.

A key strategy is full automation via package hooks. As shown below, the `caption` package allows us to define a custom format that automatically applies our `\layoutmark` to every figure and table caption throughout the document. This approach requires no manual changes to the individual `\caption` commands in the document body.

Figure/Table Captions

```
\usepackage{caption}

\DeclareCaptionFormat{marked}{
    \layoutmark{#1#2#3}
}
\captionsetup[figure]{format=marked}
\captionsetup[table]{format=marked}
```

In other cases, packages offer options that ensure more accurate bounding boxes for complex structures. Manually wrapping a multi-line mathematical environment, for instance, can be unreliable. The `empheq` package solves this by providing a `box` option that we can hook to our `\layoutmark` command, guaranteeing a single, precise bounding box around the entire environment.

Math Equations

```
\usepackage{empheq}

\begin{empheq}[box=\layoutmark]{align}
  [math content]
\end{empheq}
```

3. Command Redefinition. For common commands that lack package hooks, such as `\includegraphics`, we use LaTeX's native redefinition mechanism (`\renewcommand`). This allows us to intercept the command, wrap its content with `\layoutmark`, and then call the original command.

Command Redefinition

```
\let\oldincludegraphics
\includegraphics
\renewcommand{\includegraphics}[2][]{
  \layoutmark{
    \oldincludegraphics[#1]{#2}
}}
```

A.3 Text Line Tracker (TLT)

Tracking plain text paragraphs presents a different challenge from the discrete elements handled by NET. Paragraphs lack the explicit LaTeX markup that NET's instrumentation strategies rely on. To overcome this, the TLT leverages a novel approach inspired by the `lineno` package.

The `lineno` package was designed to add visible line numbers to a document, and its internal mechanisms must therefore process the document on a line-by-line basis. We exploit this behavior by redefining one of its core internal hooks, `\MakeLineNo`. As shown in the implementation below, our new definition first calls the original command to preserve its core functionality, then uses `\zsavepos` to record the precise baseline coordinates of the current text line. It proceeds to write a rich set of layout metadata to the `.aux` file for each line, including a paragraph identifier, column number, and line width. This process is fully automatic and requires no changes to the document's paragraph content.

Text Line Tracker

```
\usepackage[switch]{lineno}
% Add line numbers in PDF
\linenumbers
% Hidden line numbers in PDF
\renewcommand{\thelinenumber}{}

\makeatletter
% Backup the original \MakeLineNo definition.
\let\old@MakeLineNo\MakeLineNo

% Redefine \MakeLineNo to track line-level metadata.
\def\MakeLineNo{%
    % Call the original line-numbering command.
    \old@MakeLineNo

    % Save the positional metadata of the current line.
    \zsavepos{text-\the\c@linenumber}%

    % Record the current line width.
    \edef\current@linewidth{\the\linewidth}%

    % Write the metadata to the auxiliary file, including:
    % - Page number, environment type, line number, and various widths.
    \protected@write\@auxout{}{%
    \string\newlabel{line-\the\c@linenumber}{{
    page=\thepage, % Current page number.
    type=\@currenvir, % Current LaTeX environment type.
    line=\on@line, % Line number.
    linewidth=\current@linewidth, % Line width.
    textwidth=\the\textwidth, % Total text width.
    columnwidth=\the\columnwidth}}% % Column width (if applicable).
    }%

    % Call the original \MakeLineNo again for better execution.
    \old@MakeLineNo
}
\makeatother
```

A.4 Compiler Output Format

NET Compiler Output Format After compilation, each annotated element emits layout metadata to the `.aux` file in the following structured format:

NET Output Format

```
\newlabel{width}{{box width}}
\newlabel{height}{{box height}}
\newlabel{depth}{{box depth}}
\zref@newlabel{base}{
    \posx{box x}\posy{box y}
}
```

```
\zref@newlabel{element type}{
    \default{}\page{page}
}
```

This metadata provides baseline coordinates and the full dimensions of the bounding box. These values are used in subsequent stages to compute a precise layout.

TLT Compiler Output Format. The metadata for each line is emitted in the following format:

TLT Output Format

```
\zref@newlabel{text}{
    \posx{x}\posy{y}
}
\newlabel{line}{{
    page=page_id,
    type=document,
    para_id=tex_line,
    linewidth=linewidth
}}
\@LN@col{column number}
```

B Dataset Details

This section provides detailed statistics for the datasets used in our experiments. We present the distribution of element categories and the complexity of page layouts for our real-world training set, our test set, and our synthetically augmented dataset.

B.1 Real-World Training Dataset

Our real-world training dataset, comprising 20K pages, forms the foundation for our experiments. As shown in Table B.1a, the dataset exhibits a long-tail distribution typical of academic documents. Table B.1b details the layout complexity, measured by the number of unique element types per page. Most pages contain between two and five distinct element types, with a total of 3,006 unique page-level layouts observed across the corpus.

B.2 Test Dataset

Our 1K-page test set was curated to ensure a fair and realistic evaluation. As detailed in Tables B.2a and B.2b, the distribution of both individual element categories and layout complexity closely mirrors that of the real-world training set. This confirms that our evaluation is conducted on a representative, in-distribution sample of authentic documents.

B.3 Synthetic Augmentation Dataset

To address the class imbalance and limited layout variety of the real-world data, we generated a large-scale synthetic dataset of 120K pages. The statistics demonstrate two key benefits:

- **Improved Class Balance:** Table B.3a shows that our augmentation process re-balances the class distribution. The proportion of the dominant `text` class is reduced from 38.4% to 24.6%, while the representation of critical minority classes like `caption` and `footnote`.
- **Enhanced Layout Diversity:** As shown in Table B.3b, our synthetic data significantly enhances layout variety. The number of unique page-level element combinations more than doubles, from 3,006 in the real dataset to **6,126**. This ensures the model is exposed to a much broader range of structural variations, which is critical for improving robustness and generalization.

Table B.1: Statistical details of the 20K real-world training dataset. Table (a) shows the distribution of element categories, while Table (b) details the layout complexity, measured by the number of co-occurring element types per page.

(a) Element category distribution.

Category	Percentage
title	0.75%
abstract	0.73%
heading	19.65%
text	38.39%
figure	9.06%
figure caption	5.83%
table	5.40%
table caption	5.26%
math	9.28%
footnote	2.29%
reference	3.35%

(b) Page layout complexity.

Element Types	Unique Combinations	Percentage
1	42	17.8%
2	227	23.4%
3	634	20.4%
4	795	22.9%
5	803	9.4%
6	372	5.1%
7	124	1.0%
8	9	0.01%

Table B.2: Statistical details of the 1K-page test set.

(a) Element category distribution.

Category	Percentage
title	0.66%
abstract	0.67%
heading	18.96%
text	39.39%
figure	7.96%
figure caption	4.99%
table	4.96%
table caption	4.30%
math	14.05%
footnote	1.39%
reference	2.67%

(b) Page layout complexity.

Element Types	Unique Combinations	Percentage
1	17	19.8%
2	70	22.7%
3	125	21.5%
4	114	20.7%
5	78	8.6%
6	44	6.0%
7	8	0.8%

Table B.3: Statistical details of the 120K-page synthetic dataset.

(a) Element category distribution.

Category	Percentage
title	1.55%
abstract	1.56%
heading	15.43%
text	24.59%
figure	11.25%
figure caption	8.84%
table	10.24%
table caption	10.09%
math	10.15%
footnote	4.22%
reference	3.08%

(b) Page layout complexity.

Element Types	Unique Combinations	Percentage
1	60	12.3%
2	276	29.2%
3	1,049	16.9%
4	1,496	22.3%
5	1,802	11.5%
6	1,025	6.2%
7	368	1.5%
8	49	0.1%
9	1	0.01%

C Training and Evaluation Details

C.1 Training Procedure

We adopt Qwen-2.5-VL-3B as the backbone of our vision-language system, given its strong performance and growing adoption in recent multimodal research (Bai et al. 2025).

Image Preprocessing Qwen-2.5-VL-3B requires input images to have dimensions that are multiples of 28, and bounding box coordinates must be provided in absolute format (Bai et al. 2025; Qwen Team 2025b). To satisfy these constraints, we first render PDF pages into images at 110 DPI, striking a balance between visual clarity and processing cost. Each image is then resized to the nearest resolution divisible by 28 in both height and width. Bounding boxes are scaled accordingly to preserve alignment with the updated image dimensions, ensuring accurate spatial correspondence between visual content and annotations.

Prompt Design We follow the fine-tuning conventions established by the Qwen-LLaMA-Factory framework (hiyouga 2023; Qwen Team 2025a) to construct task-specific prompts. The model is guided to perform region-level layout parsing to detect and localize semantically meaningful document elements, and generate structured, machine-readable outputs, enabling downstream applications such as document understanding and semantic layout parsing.

Supervised Fine-Tuning Prompt

~~~
Locate and detect the following regions in the image: title, abstract, heading,
    footnote, figure, figure caption, table, table caption, math, text. Output each
    detected region's bbox coordinates in JSON format. The format of the output is:
<answer>```
    json[{
        "bbox_2d": [x1, y1, x2, y2],
        "label": "region name",
        "order": "reading order"}
    ]
```</answer>.
~~~

Supervised Fine-Tuning Procedure We fine-tune only the language model component while keeping the vision encoder frozen, as the visual backbone already provides strong perceptual features; thus, the goal of fine-tuning is to align these features with the expected structured outputs.

We adopt Low-Rank Adaptation (LoRA) (hiyouga 2023; Qwen Team 2025a) with rank 64, scaling factor 128, and dropout rate 0.1. The training dataset includes 20K real-world document images and 120K synthetic samples, sorted by layout complexity in ascending order. Training is conducted over 50 hours using four NVIDIA 4090D GPUs.

C.2 Evaluation Pipeline

Confidence and Non-maximum Suppression The VLM is provided with a PDF page and a structured prompt to predict a set of bounding boxes, semantic classes, and the logical reading order of elements on the page. Unlike traditional object detectors such as Faster R-CNN or YOLO, the VLM does not directly output confidence scores alongside its detections. Instead, the confidence of each prediction is inherently represented by the probability of generating the corresponding tokens. In this context, the logits produced by the VLM reflect a probability distribution over the vocabulary at each output step. Therefore, we propose to extract the probability of the predicted class token as its confidence score. This approach naturally ties the confidence to the generative process itself and provides a unified way to quantify confidence across different elements.

To finalize detections, we apply *class-wise non-maximum suppression (NMS)* based on these confidence scores. For each semantic class, we first rank all candidate detections by their confidence and then iteratively suppress overlapping boxes with an Intersection-over-Union (IoU) above a predefined threshold, retaining only the most reliable detections. This procedure effectively merges overlapping predictions while preserving high-confidence detections.

Metrics for Ordering and Grounding Following previous works (Zhao et al. 2024b; Chen et al. 2024), we evaluate both the accuracy of layout detection and the correctness of the predicted reading order using two complementary metrics:

Average Precision (AP): AP evaluates the model's ability to accurately detect and classify layout elements by measuring the overlap between predicted and ground-truth bounding boxes. We compute AP following the standard object detection protocol: for each layout class, predictions are matched to ground-truth boxes based on an Intersection over Union (IoU) threshold of 0.5. A prediction is considered correct if its IoU with a ground-truth box exceeds the threshold and the predicted class matches the ground-truth label. We then compute precision and recall across all predictions and calculate the area under the precision-recall curve. Higher AP indicates more accurate and consistent detection of layout components across the dataset.

Kendall's Tau (τ): Kendall's Tau (Lapata 2006) assesses the correspondence between the predicted reading order and the groundtruth order. For each page, we compare all pairs of elements to determine whether their relative order is preserved. The score is defined as:

$$\tau = \frac{N_{concordant} - N_{discordant}}{N_{concordant} + N_{discordant}}$$

Here, $N_{\text{concordant}}$ counts the number of element pairs whose predicted order agrees with ground truth; $N_{\text{discordant}}$ counts the mismatches. A score of $+1$ indicates perfect agreement, 0 denotes random order, and -1 reflects complete inversion. This metric directly assesses the model's ability to capture logical reading flow.

D Additional Analysis Details

D.1 Comparison with Specialized Detectors

To contextualize our VLM's performance, we also trained a specialized object detector, YOLOv8, on the same datasets. Our experiments show that on the specific task of element grounding, YOLOv8 consistently achieves higher mAP50 scores than our fine-tuned VLM. This advantage is expected and largely attributable to YOLOv8's architecture, which incorporates powerful inductive biases optimized for object detection, such as direct bounding box regression and dense spatial feature extraction.

However, this narrow focus on detection comes at the cost of versatility. Our VLM, while slightly less performant on pure grounding, demonstrates a suite of capabilities that highlight its potential for more comprehensive document intelligence tasks. Full training details and results for the YOLO are available in Appendix E.

- **Multi-Task Capability:** The VLM is inherently multi-modal and multi-task. Unlike YOLOv8, it simultaneously performs both element grounding and reading order prediction in a single forward pass.
- **Visual Robustness:** Our VLM exhibits greater resilience to visual perturbations. As shown in Appendix D.2 and F, its performance remains stable on inputs with noise like shadows or blur, conditions under which specialized detectors can fail.
- **Zero-Shot Generalization:** A significant advantage of our fine-tuned VLM over specialized detectors is its capacity for zero-shot generalization to unseen categories. This capability is most apparent before the model has over-specialized (i.e., overfit) to the training distribution. Figure D.1 provides a compelling example, where the model accurately localizes an *Algorithm* block—a category it never encountered during training—generating a correct bounding box, albeit with low confidence.

  We attribute this powerful capability to the fundamental division of labor within the VLM architecture. The pre-trained vision encoder acts as a universal feature extractor, identifying all salient visual patterns on the page, including those of unseen elements. The role of SFT is then to teach the language model how to align textual labels from a prompt (e.g., "figure", "title") with these pre-existing visual features. Essentially, the model learns the general task of "grounding a concept to a region" rather than simply memorizing a fixed set of classes. This allows it to leverage its semantic understanding to reason about and localize novel elements, a flexibility far beyond the reach of rigid, category-fixed detectors.

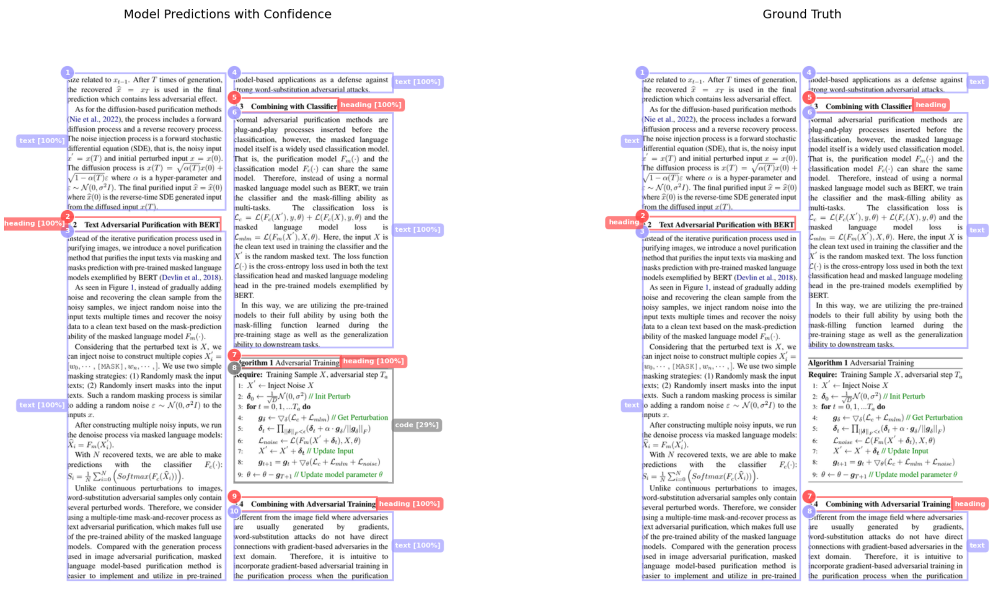
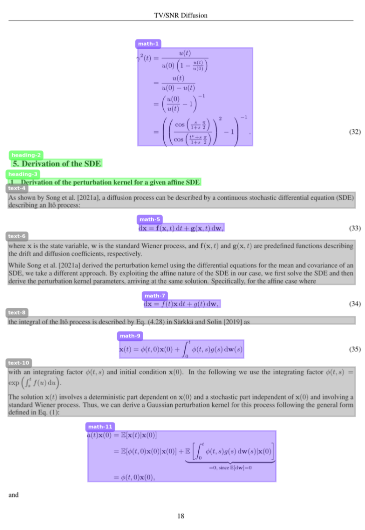


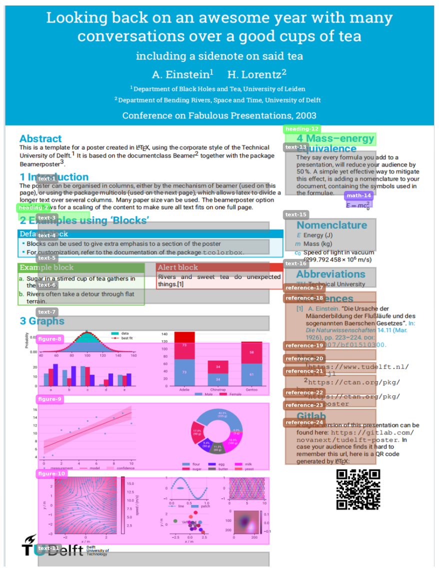
Figure D.1: Example of zero-shot generalization from our fine-tuned VLM.

Ultimately, the choice of model represents a trade-off: YOLOv8 offers superior speed and accuracy for single-task object grounding, while our VLM provides a more holistic, robust, and adaptable solution for multifaceted document understanding.

D.2 Error Analysis

Despite its strong overall performance, our fine-tuned VLM has several limitations that highlight promising avenues for future research. We discuss these below. Figure D.2 displays three representative failure cases for our VLM, while Figure D.3 shows the corresponding outputs from YOLOv8 for direct comparison.


(a) Omission of small objects


(b) Generalization to OOD layouts

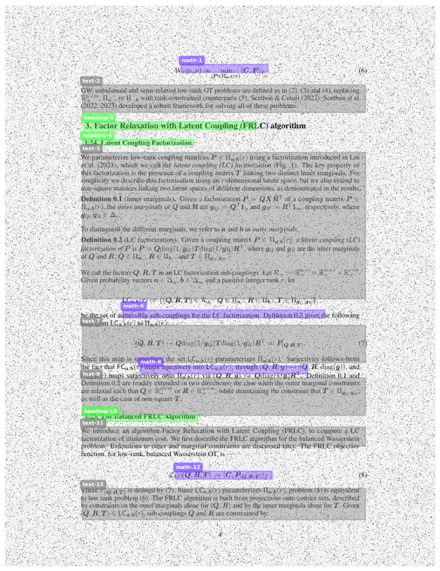
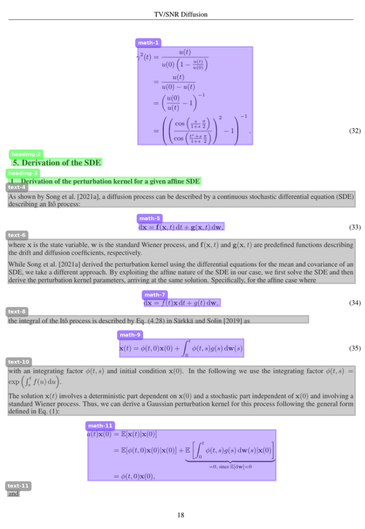
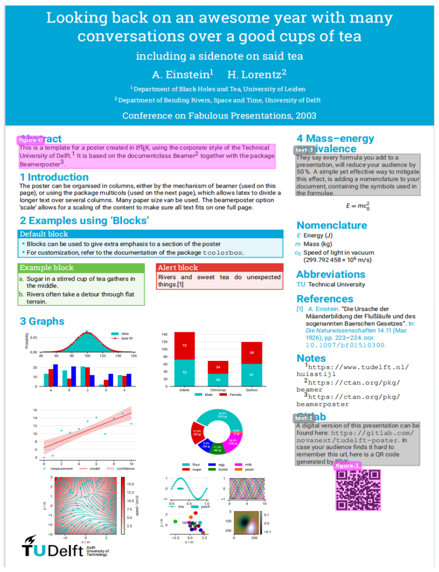

(c) Failure on severe perturbation

Figure D.2: Representative error cases for our fine-tuned VLM. (a) The model fails to detect a very small text line at the bottom of the page. (b) When presented with an out-of-distribution poster layout, the model's performance degrades. (c) Severe blurring causes detection failures.


(a) Small object detection


(b) OOD layout detection

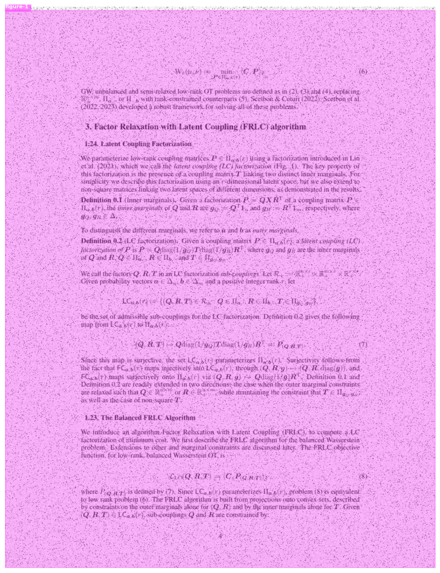

(c) Severe perturbation detection

Figure D.3: Corresponding outputs from the YOLOv8 model on the same samples. This visual comparison highlights the trade-offs between the two models: YOLOv8 is more precise on clean, in-distribution data (a), but the VLM is more robust, as YOLOv8 fails completely on the heavily blurred image (c) and perceives fewer semantic categories on the OOD poster (b).

Omission of Small Objects. Our model occasionally fails to detect very small objects, particularly single lines of text containing only a few characters. This is a classic long-tail problem, likely stemming from the low frequency of such instances in our training data, which provides an insufficient learning signal. To mitigate this, future work could pursue two complementary paths: (1) refining the training objective by up-weighting the loss for small objects, and (2) leveraging our LaTeX2Layout pipeline to programmatically generate a synthetic dataset specifically enriched with these challenging small-text elements.

Generalization to Out-of-Distribution (OOD) Layouts. A known characteristic of SFT is its tendency to specialize a model to the training distribution. Our analysis confirms this effect: while SFT makes our VLM highly proficient on academic articles, it concurrently limits its generalization to out-of-distribution layouts, such as those in newspapers or posters. However, our main results also reveal a clear solution. We have shown that VLM performance under SFT scales directly with data volume and diversity. This indicates that the most effective path to a more universal model is to drastically expand the training corpus itself. Our LaTeX2Layout pipeline is perfectly suited for this, since it can annotate any document type for which a LaTeX source exists. OOD annotation examples are showed in Appendix H. The primary bottleneck is the scarcity of public LaTeX sources for non-academic documents. Therefore, a key avenue for future work is to use large language models (LLMs) to synthesize a massive, diverse corpus of LaTeX sources across various domains. Alternatively, reinforcement learning (RL) could be explored to train a more generalizable layout parsing policy that is less reliant on the specific styles encountered during fine-tuning.

Robustness to Severe Visual Perturbations. Like most vision models, our VLM's performance degrades under excessive visual noise, such as aggressive image compression or heavy blurring. To validate a path for improvement, we conducted experiments augmenting our training data with such visual distortions. Details are in Appendix F. The results confirm that fine-tuning with these augmentations measurably enhances the VLM's robustness, demonstrating a clear strategy for improving model resilience in real-world applications where document quality can vary.

D.3 On Confidence and Grounding Accuracy

A key challenge when adapting VLMs for grounding tasks is the potential misalignment between the model's generation confidence and its spatial accuracy. Our analysis in Table D.1 reveals that the model fine-tuned on 20K examples produces outputs with very high token-level confidence (0.92 mean), yet its grounding accuracy is moderate (0.78 mAP50).

Table D.1: Confidence scores of Qwen-SFT20K and Qwen-SFT140K across categories.

Model	Tit	Abs	Head	Txt	Math	Fig	Tab	FigCap	TabCap	Foot	Ref	Average
Qwen-SFT20K	0.9783	0.9890	0.9961	0.9976	0.9648	0.9688	0.9832	1.0000	0.9063	0.8770	0.9876	0.9226
Qwen-SFT140K	0.9883	0.9997	0.9999	0.9999	0.9976	0.9989	0.9932	1.0000	0.9883	0.9670	0.9876	0.9836

We attribute this discrepancy to the standard VLM training objective. The cross-entropy loss for next-token prediction incentivizes syntactic fluency—generating a well-formed bounding box in the correct format—but does not explicitly enforce the geometric correctness of the coordinates themselves. While scaling the dataset to 140K samples improves both confidence (0.98) and accuracy (0.91 mAP50), indicating better internal alignment, the fundamental mismatch persists. This insight highlights a critical avenue for future research: developing spatially-aware training objectives or post-hoc calibration methods to better align a VLM's prediction confidence with its true localization accuracy.

D.4 Efficiency Analysis

Table D.2: Efficiency analysis of the Latex2Layout pipeline and model inference.

Component	**Task / Description**	**Time**	**Hardware**
Dataset Generation	1. Injecting NET/TLT markers into LaTeX source	0.135 s/page	AMD Ryzen 7 5800H
	2. Compiling LaTeX source via `pdflatex`	0.244 s/page	
	3. Parsing metadata and computing bounding boxes	0.056 s/page	
	Total Generation Time	**0.435 s/page**	
Model Inference	Qwen-2.5-VL-3B-Instruct	0.925 s/element	NVIDIA RTX 4090D

We evaluate the computational efficiency of our pipeline to assess its practicality for large-scale applications. As shown in Table D.2, the entire process requires only **0.435 seconds per page** on a commodity CPU (AMD Ryzen 7 5800H), confirming its scalability for creating large-scale datasets at low cost. The inference speed of our fine-tuned VLM is 0.925 seconds per element on an NVIDIA RTX 4090D GPU. As established in our comparison with specialized detectors (Appendix D.1), we consider this a practical trade-off, balancing the model's versatility and robustness against its higher computational cost.

D.5 Training Loss Comparison

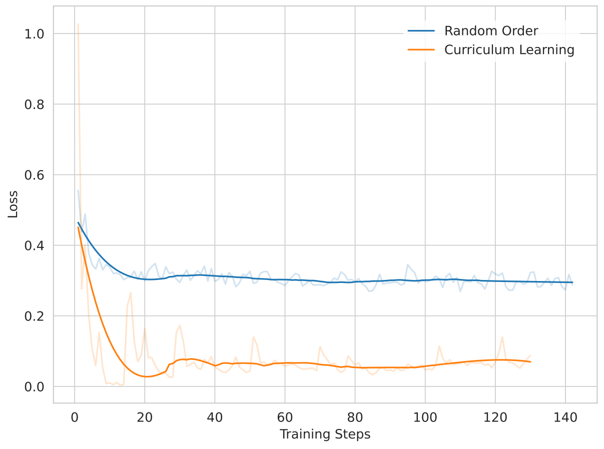


Figure D.4: Training loss comparison. The curriculum learning strategy (orange curve) converges faster and to a lower loss value than the standard random shuffling baseline (blue curve).

E YOLOv8 Training and Performance

To provide a strong baseline for the element grounding task, we trained a specialized object detector, YOLOv8, on our datasets. We adhered to a standard training protocol to ensure a fair comparison. The model was trained for 150 epochs with a batch size of 16. We used the AdamW optimizer with an initial learning rate of 1×10^{-3} and a cosine annealing schedule. Input images were resized to a resolution of 1024x1024 pixels.

The results are presented in Table E.1. As expected, the YOLOv8 model, with its architecture highly optimized for detection, achieves excellent grounding performance. The model trained on the full 140K dataset demonstrates superior accuracy across nearly all categories compared to the one trained on only 20K real samples, underscoring the benefit of our large-scale dataset even for specialized detectors.

Table E.1: Performance of YOLOv8-L on the element grounding task. As a vision-only detector, it cannot perform reading order prediction. Best scores are shown in **bold**.

Model	Reading Order (τ)	Element Grounding (AP50)											
		Tit	Abs	Head	Txt	Math	Fig	Tab	FigCap	TabCap	Foot	Ref	Average
YOLOv8 (20K Real)	–	0.95	0.95	0.92	0.96	0.87	0.94	0.97	0.91	0.91	0.85	0.97	0.93
YOLOv8 (140K Combined)	–	**0.97**	**0.96**	**0.94**	**0.97**	**0.91**	**0.96**	**0.98**	**0.94**	**0.94**	**0.88**	**0.98**	**0.95**

F Optimization for Visual Perturbation

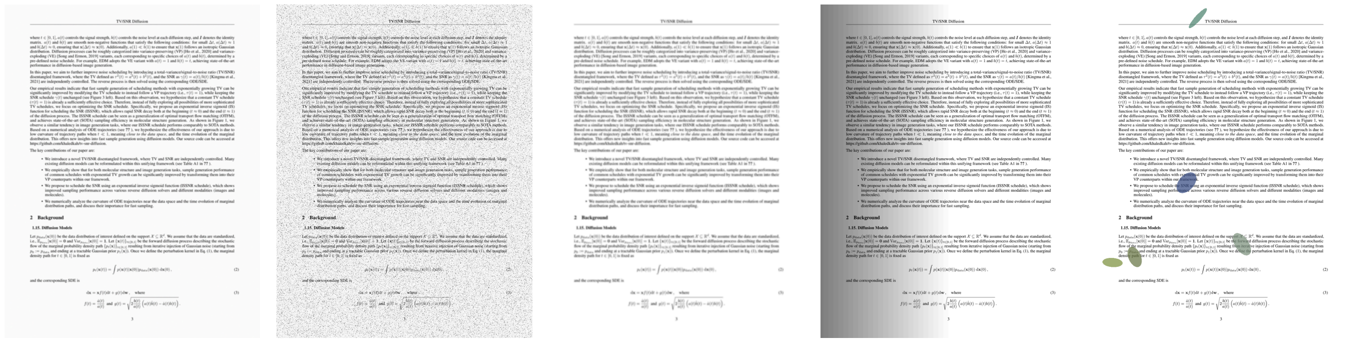
To evaluate and enhance our VLM's resilience to real-world document degradation, we conducted a targeted data augmentation experiment. This section details our augmentation strategy and presents a visual comparison and analysis.

F.1 Data Augmentation and Fine-Tuning

We created an augmented version of our 20K real-world dataset by applying a range of visual perturbations that simulate common document quality issues. For each image in the real-world set, one of the following five augmentations was randomly selected and applied:

- **Scanning Noise:** Simulates the effect of a document scanner by adding subtle Gaussian noise and a slight blur.
- **Blurring:** Applies a more significant Gaussian blur to mimic an out-of-focus effect.
- **Ink Effects:** Introduces salt-and-pepper noise and varies brightness to simulate inconsistent ink application.
- **Shadows:** Applies a gradient mask to create the appearance of uneven lighting.
- **Stains:** Adds semi-transparent elliptical blotches to simulate stains or water damage.

Visual examples of these transformations are provided in Figure F.1. We then used this newly created ”noisy” dataset for a second stage of SFT, starting from the checkpoint of the model already trained on the clean 140K dataset. This process yielded our enhanced, noise-robust model, hereafter referred to as VLM-Aug.



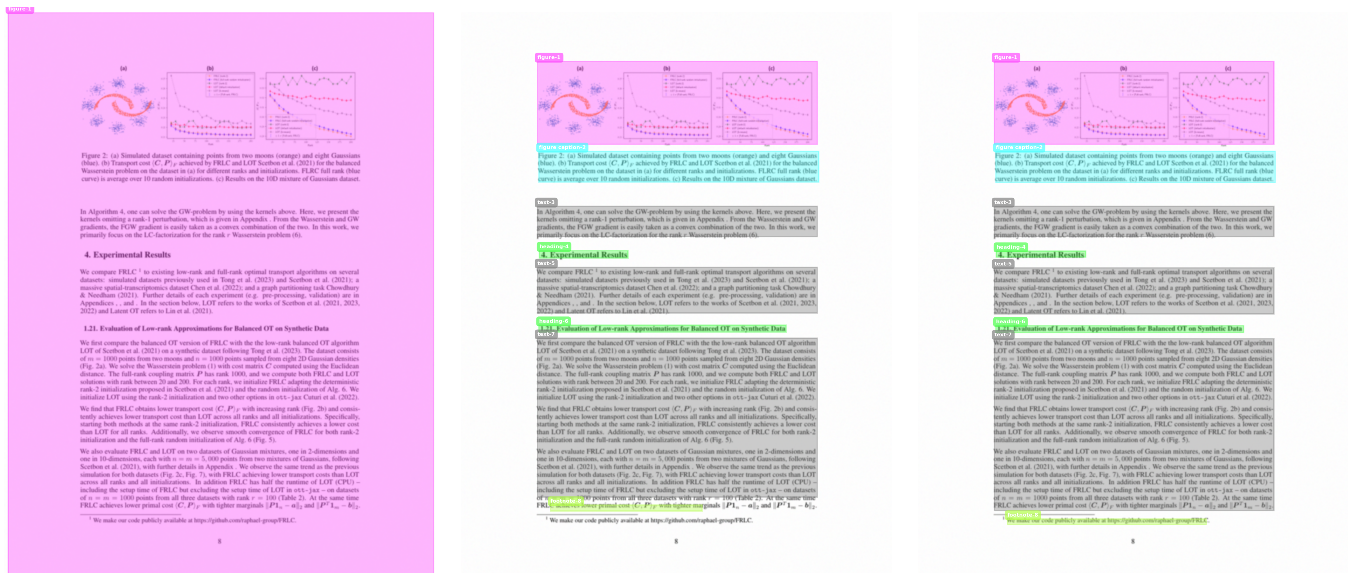
(a) Scanning Noise (b) Blurring (c) Ink Effects (d) Shadows (e) Stains

Figure F.1: Examples of the five visual perturbation types randomly applied to our 20K real-world dataset to create the augmented training set.

F.2 Visual Comparison and Analysis

Figure F.2 presents a side-by-side comparison of model outputs on a heavily blurred test page. This visual evidence confirms two key findings. First, our base VLM possesses greater intrinsic robustness to visual noise than the specialized detector. The trained YOLOv8 model (Figure F.2a) fails catastrophically on this perturbed input, misclassifying the entire page as a single `figure` element. In contrast, our original VLM (Figure F.2b) demonstrates significant inherent robustness. It correctly identifies the main structural elements of the page, struggling only with the precise localization of the small `footnote`.

Second, this inherent robustness can be significantly enhanced. After the second stage of fine-tuning on the augmented data, our VLM-Aug model (Figure F.2c) correctly parses the entire layout, accurately localizing the footnote. This experiment validates that targeted data augmentation is a highly effective strategy for creating VLMs that are resilient to the visual degradation common in real-world documents.



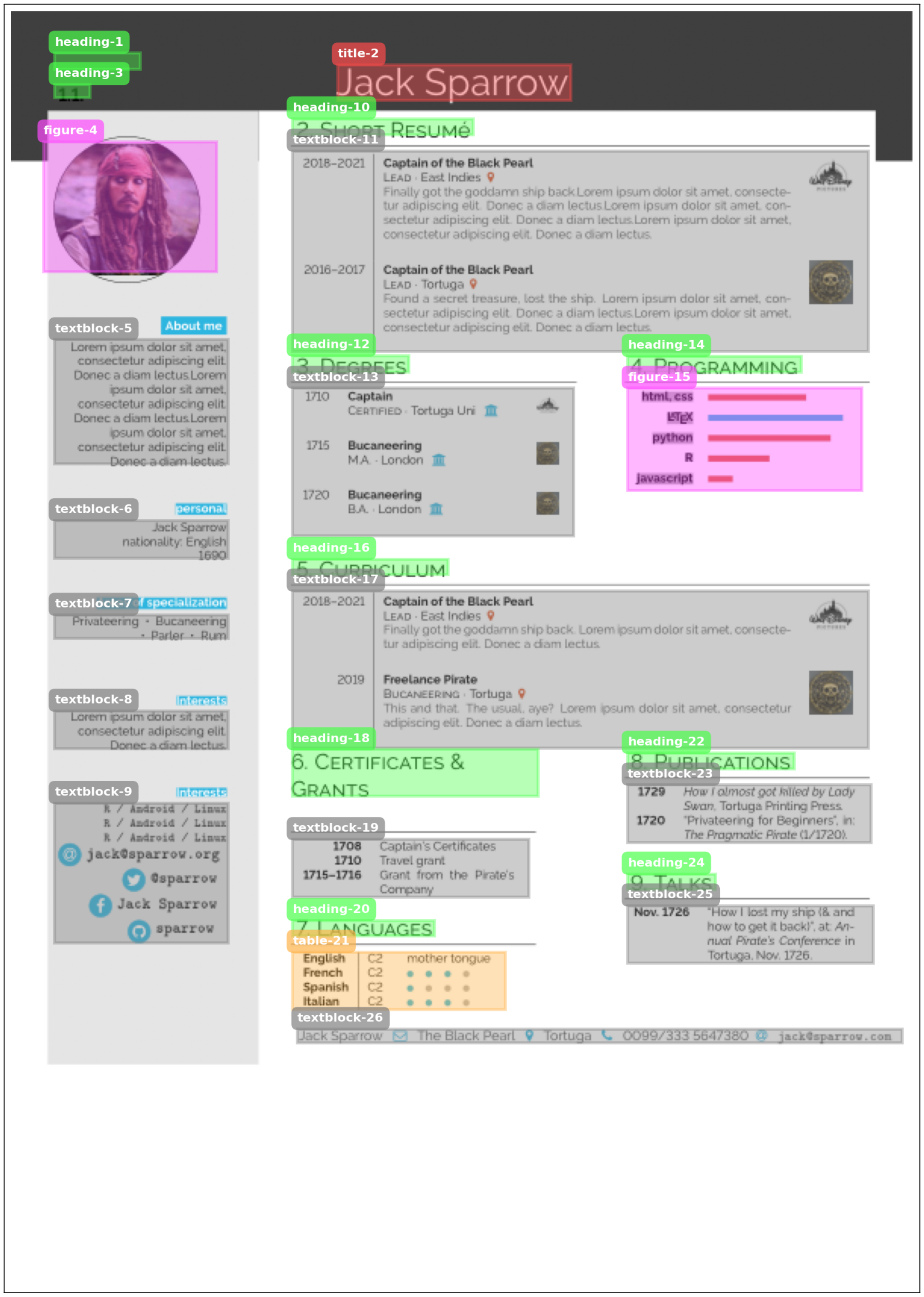
(a) YOLOv8 (trained on 140K clean data) (b) Original VLM (SFT on 140K clean data) (c) Augmented VLM (SFT on 140K clean data + SFT on 20K noisy data)

Figure F.2: A visual comparison of model performance on a heavily blurred document. (a) YOLOv8 fails completely. (b) Our original VLM correctly identifies the main layout but misplaces the footnote. (c) The VLM fine-tuned on augmented data correctly parses the entire page.

G More Examples of Synthetic Data Annotations

Figure G.1: Examples of our synthetic dataset.

H Latex2Layout Generalization Examples



1.1

Jack Sparrow

2. Short Resumé

2018–2021 **Captain of the Black Pearl**
Lead · East Indies
Finally got the goddamn ship back.Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.

2016–2017 **Captain of the Black Pearl**
Lead · Tortuga
Found a secret treasure, lost the ship. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.

About me

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.

3. Degrees

1710 **Captain**
Certified · Tortuga Uni

1715 **Bucaneering**
M.A. · London

1720 **Bucaneering**
B.A. · London

4. Programming

personal

Jack Sparrow
nationality: English
1690

5. Curriculum

2018–2021 **Captain of the Black Pearl**
Lead · East Indies
Finally got the goddamn ship back. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.

2019 **Freelance Pirate**
Bucaneering · Tortuga
This and that. The usual, aye? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.

of specialization

Privateering · Bucaneering · Parler · Rum

Interests

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec a diam lectus.

Interests

```
R / Android / Linux
R / Android / Linux
R / Android / Linux
jack@sparrow.org
@sparrow
Jack Sparrow
sparrow
```

6. Certificates & Grants

1708	Captain's Certificates
1710	Travel grant
1715–1716	Grant from the Pirate's Company

7. Languages

English	C2	mother tongue
French	C2	● ● ● ○
Spanish	C2	● ○ ○ ○
Italian	C2	● ● ● ○

8. Publications

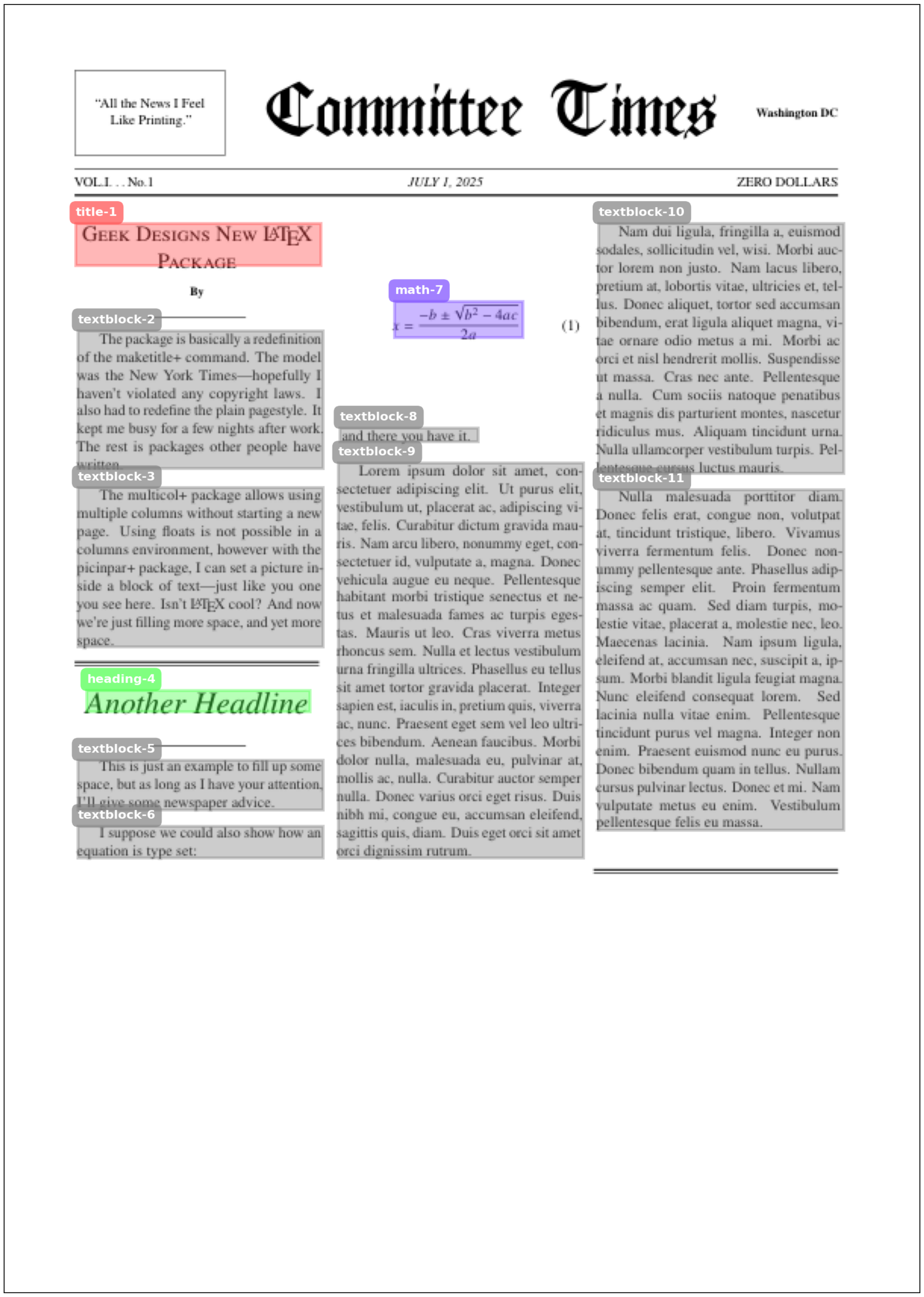
1729	*How I almost got killed by Lady Swan*, Tortuga Printing Press.
1720	"Privateering for Beginners", in: *The Pragmatic Pirate* (1/1720).

9. Talks

Nov. 1726	"How I lost my ship (& and how to get it back)", at: *Annual Pirate's Conference* in Tortuga, Nov. 1726.

Jack Sparrow · The Black Pearl · Tortuga · 0099/333 5647380 · jack@sparrow.com

Figure H.1: Resume



"All the News I Feel Like Printing."

Committee Times

Washington DC

VOL.I. . . No.1 — *JULY 1, 2025* — ZERO DOLLARS

GEEK DESIGNS NEW LATEX PACKAGE

By

The package is basically a redefinition of the maketitle+ command. The model was the New York Times—hopefully I haven't violated any copyright laws. I also had to redefine the plain pagestyle. It kept me busy for a few nights after work. The rest is packages other people have written.

The multicol+ package allows using multiple columns without starting a new page. Using floats is not possible in a columns environment, however with the picinpar+ package, I can set a picture inside a block of text—just like you one you see here. Isn't LaTeX cool? And now we're just filling more space, and yet more space.

Another Headline

This is just an example to fill up some space, but as long as I have your attention, I'll give some newspaper advice.

I suppose we could also show how an equation is type set:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{1}$$

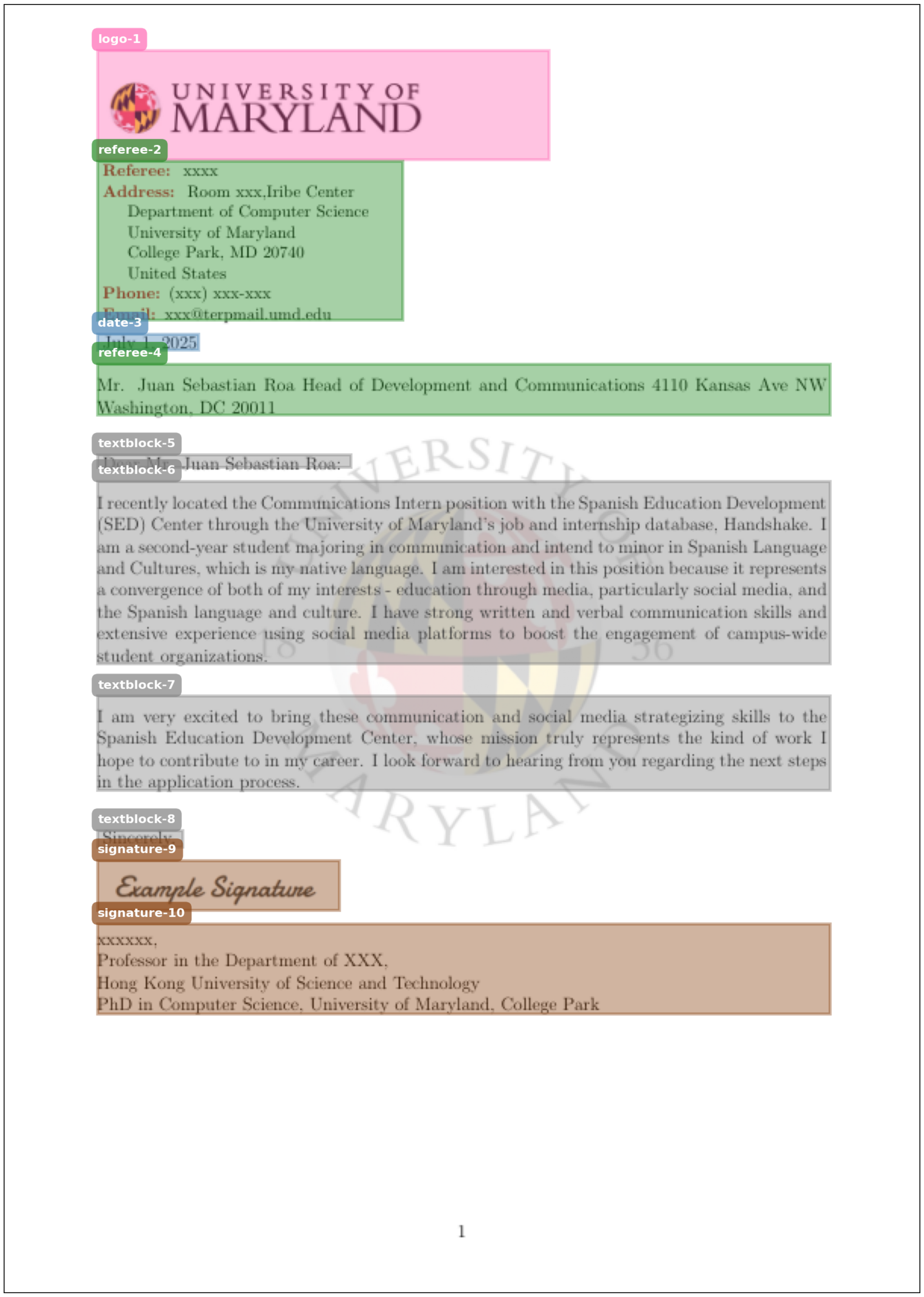
and there you have it.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Figure H.2: Newspaper


logo-1
UNIVERSITY OF MARYLAND
referee-2
date-3
referee-4
textblock-5
textblock-6
textblock-7
textblock-8
signature-9
signature-10

Referee: xxxx
Address: Room xxx,Iribe Center
Department of Computer Science
University of Maryland
College Park, MD 20740
United States
Phone: (xxx) xxx-xxx
Email: xxx@terpmail.umd.edu

July 1, 2025

Mr. Juan Sebastian Roa Head of Development and Communications 4110 Kansas Ave NW Washington, DC 20011

Dear Mr. Juan Sebastian Roa:

I recently located the Communications Intern position with the Spanish Education Development (SED) Center through the University of Maryland's job and internship database, Handshake. I am a second-year student majoring in communication and intend to minor in Spanish Language and Cultures, which is my native language. I am interested in this position because it represents a convergence of both of my interests - education through media, particularly social media, and the Spanish language and culture. I have strong written and verbal communication skills and extensive experience using social media platforms to boost the engagement of campus-wide student organizations.

I am very excited to bring these communication and social media strategizing skills to the Spanish Education Development Center, whose mission truly represents the kind of work I hope to contribute to in my career. I look forward to hearing from you regarding the next steps in the application process.

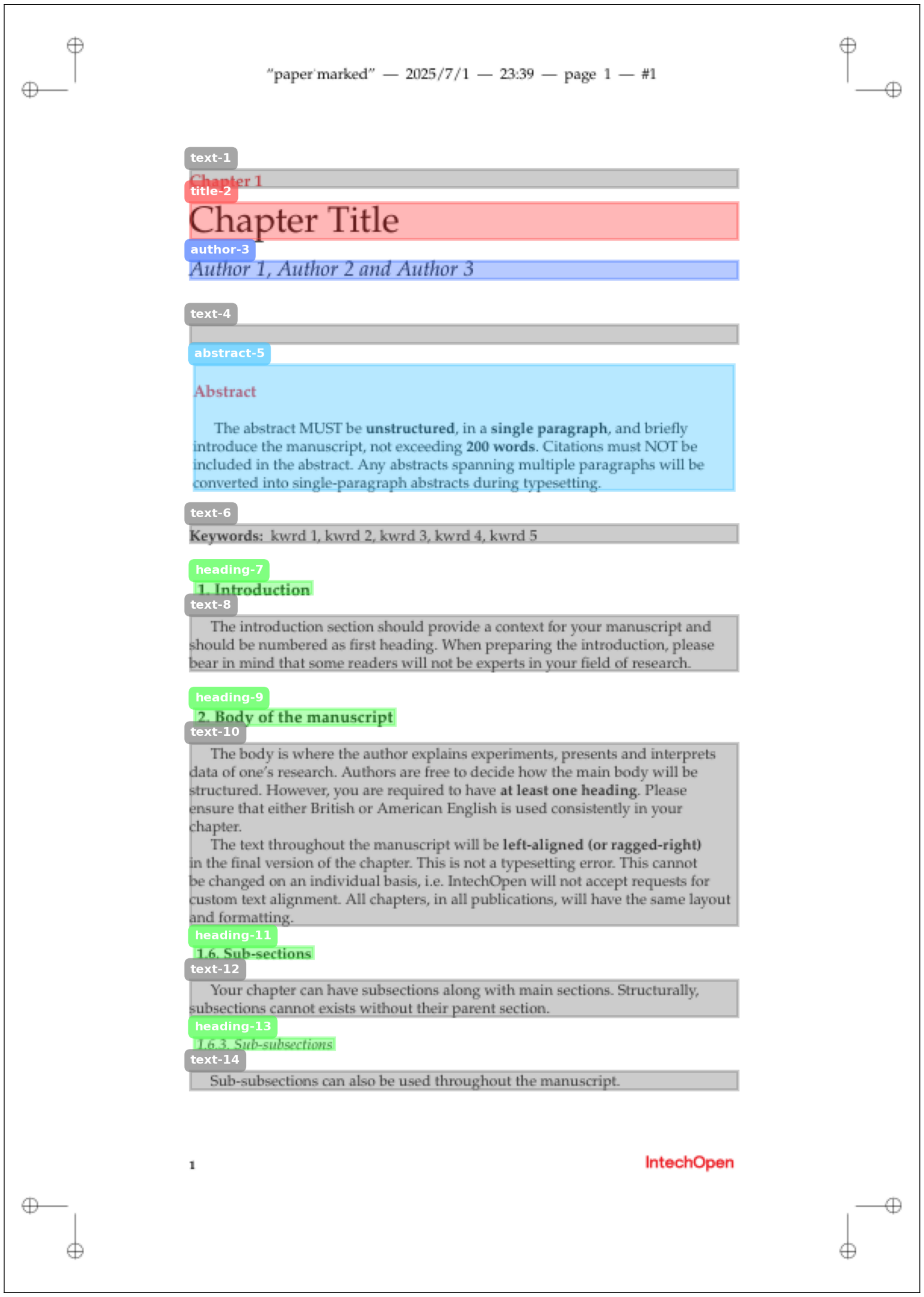
Sincerely,

Example Signature

xxxxxx,
Professor in the Department of XXX,
Hong Kong University of Science and Technology
PhD in Computer Science, University of Maryland, College Park

1

Figure H.3: Cover Letter



"paper marked" — 2025/7/1 — 23:39 — page 1 — #1

Chapter 1

Chapter Title

Author 1, Author 2 and Author 3

Abstract

The abstract MUST be **unstructured**, in a **single paragraph**, and briefly introduce the manuscript, not exceeding **200 words**. Citations must NOT be included in the abstract. Any abstracts spanning multiple paragraphs will be converted into single-paragraph abstracts during typesetting.

Keywords: kwrd 1, kwrd 2, kwrd 3, kwrd 4, kwrd 5

1. Introduction

The introduction section should provide a context for your manuscript and should be numbered as first heading. When preparing the introduction, please bear in mind that some readers will not be experts in your field of research.

2. Body of the manuscript

The body is where the author explains experiments, presents and interprets data of one's research. Authors are free to decide how the main body will be structured. However, you are required to have **at least one heading**. Please ensure that either British or American English is used consistently in your chapter.

The text throughout the manuscript will be **left-aligned (or ragged-right)** in the final version of the chapter. This is not a typesetting error. This cannot be changed on an individual basis, i.e. IntechOpen will not accept requests for custom text alignment. All chapters, in all publications, will have the same layout and formatting.

1.6. Sub-sections

Your chapter can have subsections along with main sections. Structurally, subsections cannot exists without their parent section.

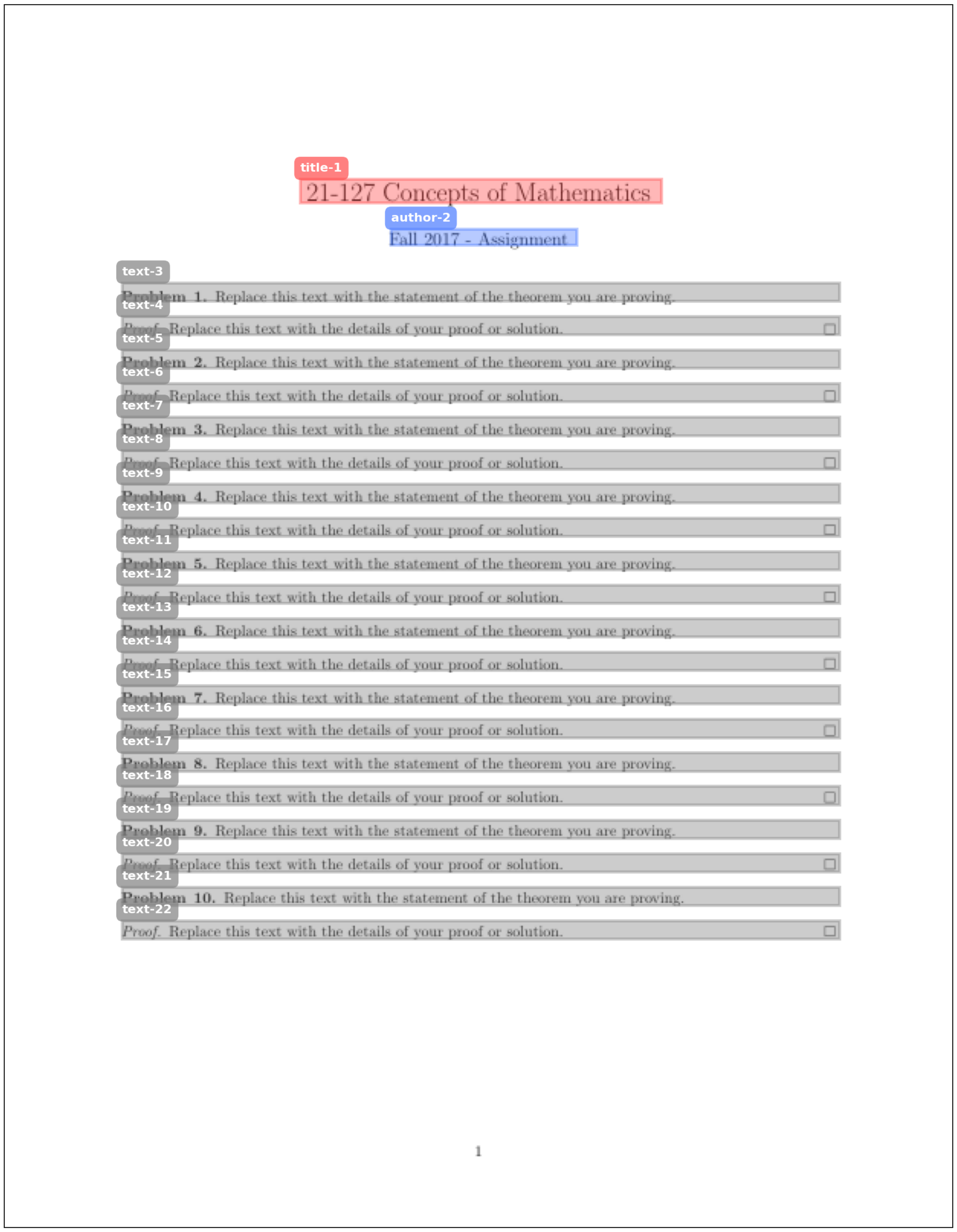
1.6.3. Sub-subsections

Sub-subsections can also be used throughout the manuscript.

1

IntechOpen

Figure H.4: TextBook



21-127 Concepts of Mathematics

Fall 2017 - Assignment

Problem 1. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 2. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 3. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 4. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 5. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 6. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 7. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 8. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 9. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

Problem 10. Replace this text with the statement of the theorem you are proving.

Proof. Replace this text with the details of your proof or solution. □

1

Figure H.5: Homework