iTagPDF: Towards Finally Automating PDF Accessibility

Peya Mowar
Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
pmowar@andrew.cmu.edu

Aaron Steinfeld
Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
steinfeld@cmu.edu

Jeffrey P Bigham
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
jbigham@cs.cmu.edu

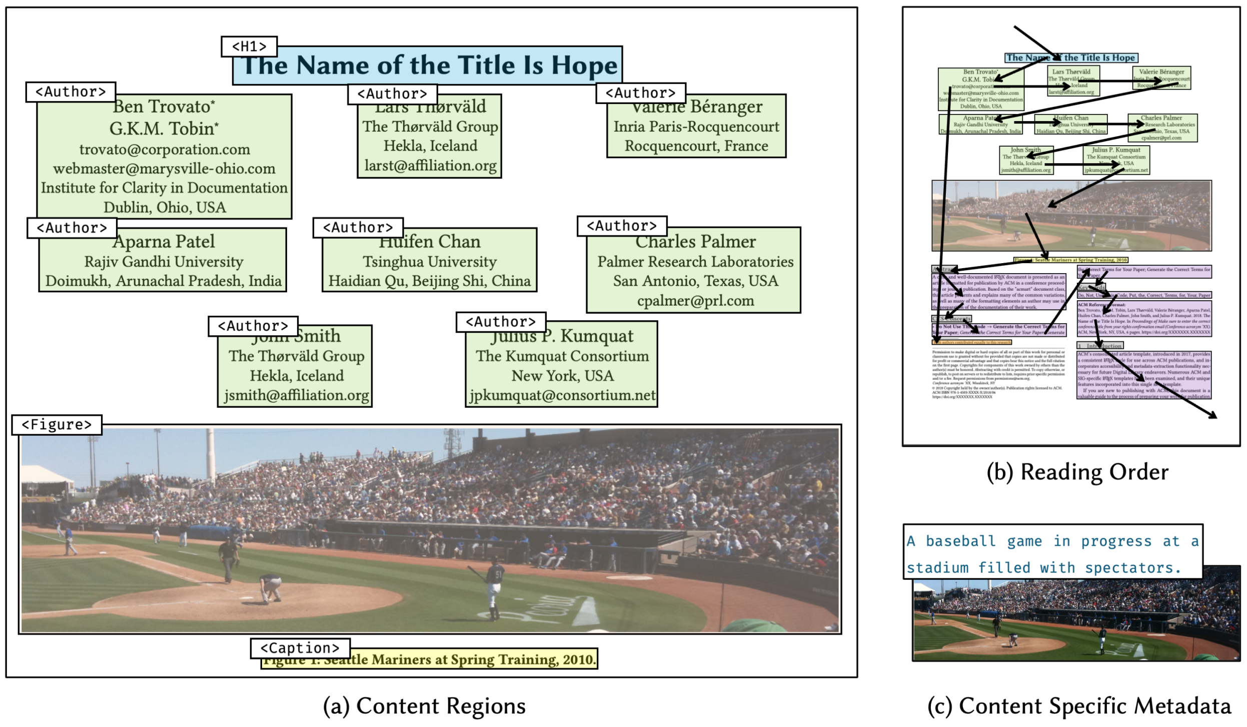

<H1>
The Name of the Title Is Hope
<Author>
<Figure>
<Caption>
Figure 1: Seattle Mariners at Spring Training, 2010.
(a) Content Regions
(b) Reading Order
A baseball game in progress at a stadium filled with spectators.
(c) Content Specific Metadata

Figure 1: Intelligent PDF Tagging (iTagPDF) automatically produces three kinds of PDF accessibility metadata: (a) content regions and their locations (tags), (b) their reading order, and (c) content specific metadata, such as structure information for tables and alt texts for figures and formulas. iTagPDF is the first to combine all major tasks of making a PDF accessible into one tool, and outperforms all prior approaches by jointly modeling the input source and the rendered pixels of the PDF.

Abstract

Most academic research is ultimately disseminated through documents in the PDF format. This format has advantages in flexibility and portability, but presents challenges for accessibility that have stubbornly resisted solutions despite decades of attempts. Tagging PDFs is hard to automate because tags are currently generated visually, not semantically, which makes the output cluttered and manual correction tedious and error-prone. Ironically, this semantic structure already exists during authoring but is discarded during PDF rendering. This raises an obvious question, can we use this lost semantic information to better automate tagging in PDFs? In this paper, we develop **iTagPDF**, a system that refines generated metadata using the semantics in the source documents of research papers. We demonstrate that the metadata generated by iTagPDF already surpasses what authors currently submit to ACM conferences on many criteria. Our approach represents a concrete step toward finally automating accessibility remediation in research paper PDFs.

ACM ISBN 979-8-4007-2278-3/26/04
https://doi.org/10.1145/3772318.3790289

CCS Concepts

• Human-centered computing → Accessibility systems and tools; **• Computing methodologies →** *Computer vision tasks*; **• Applied computing → Document metadata**.

ACM Reference Format:
Peya Mowar, Aaron Steinfeld, and Jeffrey P Bigham. 2026. iTagPDF: Towards Finally Automating PDF Accessibility. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3772318.3790289

1 Introduction

Most academic research is disseminated through documents in the Portable Document Format (PDF), primarily because this format preserves formatting across many different platforms. Originally a proprietary printer format introduced by Adobe, PDF has evolved significantly over the decades into a versatile and flexible format with an open standard. Despite this, persistent challenges remain in ensuring that documents prepared in this format consistently include the metadata necessary to make them accessible to people with disabilities [2, 3]. These challenges are inextricably tied to the complicated and confusing procedures required to make a PDF accessible. The PDF format is used almost exclusively in the academic community despite its shortcomings.

Making a PDF accessible requires three primary kinds of metadata, which directly translate to how assistive technology conveys the content: *(i)* tagging individual sections of the PDF content according to their type and location on the page (*e.g.*, heading, paragraph, figure, etc.), *(ii)* defining an appropriate reading order, and *(iii)* providing content-specific metadata (*e.g.*, alternative text description for images or structured information for tables and lists).

The tools available for providing this necessary metadata (called accessibility remediation) are tedious to use and it is difficult to know if you have made the PDF accessible correctly. Although a number of tools exist for PDF remediation [1, 7, 32], they mostly work in the same way – the tool visualizes each page of the PDF document including any existing metadata information and the user must manually provide (or fix) the metadata. Because the metadata is only visible via specialized remediation tools and assistive technologies, it often goes unchecked or even accidentally removed during various processing steps during publication.

Given the general difficulty of working with PDFs, several projects have instead approached the problem as one of transforming the PDF into formats that are easier to work with, such as HTML. As an example, one goal of the ACM TAPS[1] process is to gather sufficient data from authors in order to produce papers in multiple formats, including both PDF and HTML. Despite extensive work in preparing source documents to comply with TAPS, authors still need to manually remediate their PDFs for them to be accessible. HTML versions are created as part of the ACM Digital Library, but they tend not to be as easily available or shared, such as on an offline device or authors' homepages.

[1]https://authors.acm.org/proceedings/production-information/taps-production-workflow

In this work, we explore if we can leverage the multiple representations of papers that are naturally created during the authoring process for automatic research paper PDF remediation. We build a system called iTagPDF that uses this combined representation to create highly accurate tags in the correct reading order. To evaluate iTagPDF, we create a first-of-kind dataset for academic paper PDFs that includes metrics specifically created for accessibility.

In particular, we leverage both the source document of the paper (LaTeX) and its visual rendering (PDF). The visual rendering is an important source because it is the final output that readers are assumed to have, and is the result of multiple steps during the publication process that are difficult to capture from source alone. We use the visual rendering in two ways – first, we run a state-of-the-art object detection model on the visual rendering of the PDF, and also we run optical character recognition on each segment to determine what text it contains. These steps produce two sets of visual bounding boxes to ground the location of the text visually on the page. We then use the text as an index into the source document, which we use to find related metadata (*e.g.*, heading levels, alt texts), and also to influence reading order where visual layout is unclear (common in two column formats).

We believe our method is starting to approach the quality level needed to perform tagging and reading order remediation automatically. While we advise being appropriately cautious, we are not the first to recognize that AI may be starting to get to the point where it may obviate the need for human intervention and that this may allow us to finally solve long-standing problems in accessibility [36]. In the particular instance of PDF tagging and reading order, we believe this is likely to become nearly solved because the information necessary to do this well is already available when producing PDFs, it just exists in disparate sources.

In summary, this paper contributes the following:

- **iTagPDF**: An automatic PDF remediation system that combines the visual and source representations of PDFs to better detect tags, their reading order, and content specific metadata.
- A new dataset and associated metrics targeted at what matters for accessibility of PDF research papers.
- An evaluation of our novel method establishing a strong baseline for this dataset.

2 Related Work

Making academic paper PDFs accessible has remained a persistent challenge over the past decade [3, 10]. This is partly due to the technical limitations of existing PDF remediation tools, and partly due to the human challenges involved in the authoring process. We discuss these challenges and examine the emerging role of vision-language models in PDF accessibility.

2.1 Human Challenges in Authoring Accessible Research Papers

The research community overwhelmingly relies on PDF as the standard format for disseminating academic papers because of its consistency, portability, and print fidelity. However, this choice significantly undermines accessibility. To make a PDF accessible, authors must proactively supply the required accessibility metadata, *i.e.,* its underlying semantic markup, that enables assistive technologies, such as screen readers, to interpret the document's content correctly. Unsurprisingly, authors often fail to provide this metadata consistently or at adequate quality, rendering most academic content inaccessible to people with disabilities, as the following studies demonstrate.

A decade ago, Brady et *al.* [3] examined the accessibility of 1.8k PDFs published at ACM venues and found only a fraction of papers were tagged, even in conferences related to accessibility, with particularly poor coverage at CHI (<26.8%). Wang et *al.* [39] expanded this analysis through automated evaluation of roughly 11k academic PDFs from diverse venues and noted that only 2.4% papers were compliant with Adobe-5 standards. Unfortunately, these numbers have seen little improvement to this day, with Kumar et *al.* [10] recently reporting just 3.2% compliance among the ~20k papers they assesed using the Adobe-6 standards.

Lazar et *al.* [11] inspected the accessibility of multiple years of CHI proceedings and compared several intervention strategies including requesting, mandating, or distributing the responsibility for PDF remediation among authors and other stakeholders in the publishing process. They reported that even in the year when authors received considerable feedback about the accessibility issues in their submissions, the proportion of tagged PDFs only increased marginally (from 20.6% to 26.8%) and failed to sustain in the subsequent years. Bigham et *al.* [2] further detailed why authors neglected in making their documents accessible. They argued authors lacked sufficient incentives to engage with a complex remediation process, largely due to inadequate tooling. At the time, they noted that the community had already been waiting nearly a decade for better tools, and cautioned that even if such tools were eventually improved, it would remain unclear whether authors would use them effectively (or at all).

Menzies et *al.* [17] studied author perspectives on creating accessible documents and revealed many challenges in PDF remediation such as the laborious nature of the process, requirement for specialized knowledge, difficulties in verifying correctness, reliance on expensive software, and complications introduced by the publishing workflow (e.g., re-running the software for every submission, publishers inadvertently stripping metadata, and the process typically being left to the last stage before deadlines). Notably, they underscored how authors with disabilities faced additional barriers due to the inaccessibility of remediation tools themselves, which delayed their progress and limited their agency. For example, the process of visually tagging a PDF itself is inaccessible to blind authors. These findings make it clear that expecting authors to manually make their PDFs accessible is neither effective nor viable, underscoring the importance of shifting toward automated solutions.

2.2 Tooling for Making PDFs Accessible

Though making PDFs automatically accessible seems like a solvable problem, those who have worked on it have noted that it is a messy problem requiring substantial engineering effort [2]. This is because PDFs are rooted in visual formatting, which makes it difficult to extract their underlying structure. Three kinds of tools have been developed to tackle PDF accessibility: (i) borne-accessible PDF creation platforms, (ii) PDF remediation software, and (iii) PDF to HTML (or other accessible formats) conversion tools. We discuss the individual benefits and limitations of each of these categories below.

Some PDF creation platforms, such as Microsoft Word, enable authors to export an accessible PDF directly from the editor. However, this approach has two major drawbacks. First, both Wang et *al.* [39] and Kumar et *al.* [10] found that accessibility compliance varies substantially by the typesetting platform, with Microsoft Word producing the most accessible PDFs while LaTeX invariably yielding the least. Second, Rajkumar et *al.* [30] noted that even correctly generated borne-accessible PDFs often still required additional remediation.

These limitations have motivated long-standing efforts to produce borne-accessible PDFs specifically within the LaTeX ecosystem. Some packages in LaTeX (such as pdfx[2], hyperxmp[3], accessibility[4]) provide partial support for embedding metadata directly in generated PDFs but do not fully automate tagging of the document structure. The LaTeX Tagged PDF project [21] initiated in 2020 as a multi-year effort, aims to enable the generation of fully accessible PDFs using LaTeX. It seeks to overcome the limitation whereby LaTeX discards much of the document's contained semantic structure when producing a PDF, as noted by prior work [2, 21, 30]. However, even five years after its initiation, recent updates note that the project supports automatic generation of accessible PDFs for only a subset of LaTeX documents that import packages from a small pre-defined list [18, 19].

PDF remediation software, particularly Adobe Acrobat [1], has become the de facto standard for making academic PDFs accessible as it can be applied regardless of the original PDF creation process. However, as discussed in the previous subsection, prior works [2, 17] have extensively documented the challenges of using Acrobat, including the high cost of software license, an unintuitive user interface, and the low-quality of automated tags requiring authors to manually review and correct them. Prior work has assessed the usability of existing open-source tools [30] such as PAVE[5], and developed AI-supported PDF remediation tools with more intuitive user interfaces, including as the earlier Ally [29] and the latest PAVE 2.0 by Schmitt-Koopmann et *al.* [32]. However, while

[2]https://ctan.org/pkg/pdfx
[3]https://ctan.org/pkg/hyperxmp
[4]https://ctan.org/tex-archive/macros/latex/contrib/accessibility
[5]https://pave-pdf.org/?lang=en

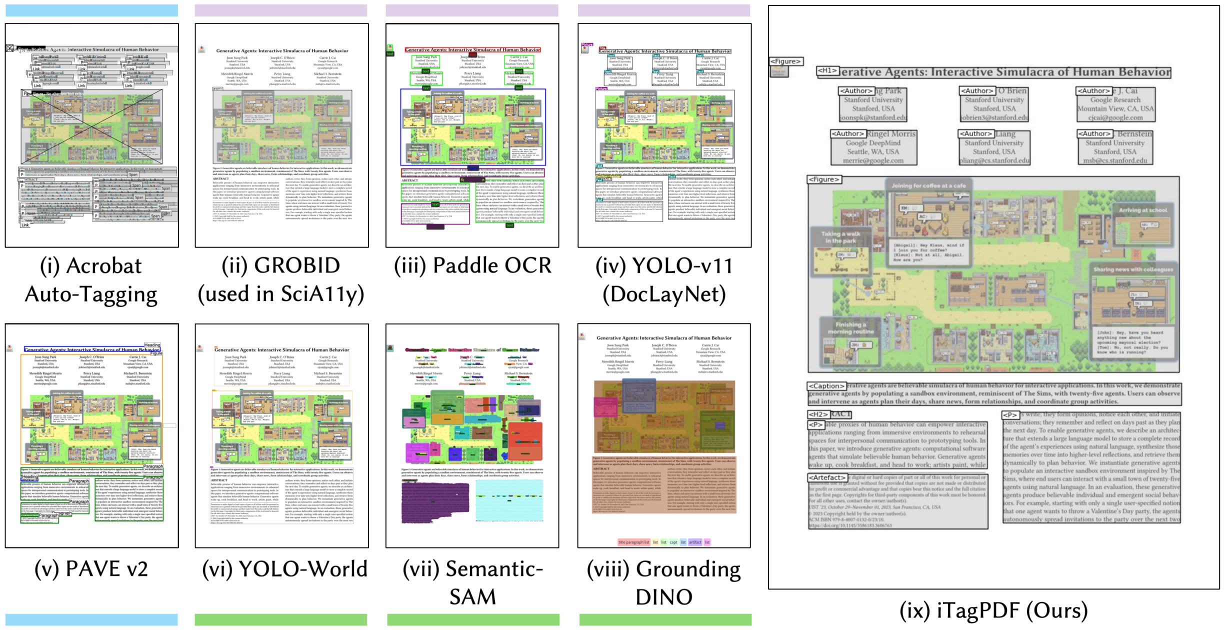


Figure 2: Visualizations of metadata generated for a research paper [27] by (a) accessibility remediation tools (i, v in blue), (b) machine learning pipelines for document layout analysis (ii–iv in lilac), (c) vision-language models for object detection (vi–viii in green), and (d) iTagPDF (ix). Existing approaches rely solely on visual features of the PDF, producing cluttered, incomplete or incorrect metadata. In contrast, the metadata identified visually by iTagPDF is further refined using additional context from source documents (e.g., LaTeX source files), resulting in cleaner and more semantically accurate results.

Ally was notably inaccessible due to the dexterity required for drawing reading order arrows with a mouse, PAVE 2.0 cannot automatically detect elements such as captions and footnotes because of limitations in the classes included in its training dataset. Schmitt-Koopmann et *al.* further disclosed that 20% of their participants were willing to spend only under a minute per page to make their PDFs accessible and longed for a fully automated approach to tagging PDFs. Nonetheless, no prior tool fully automates PDF accessibility remediation, due to the inherently intractable nature of retroactively tagging PDFs.

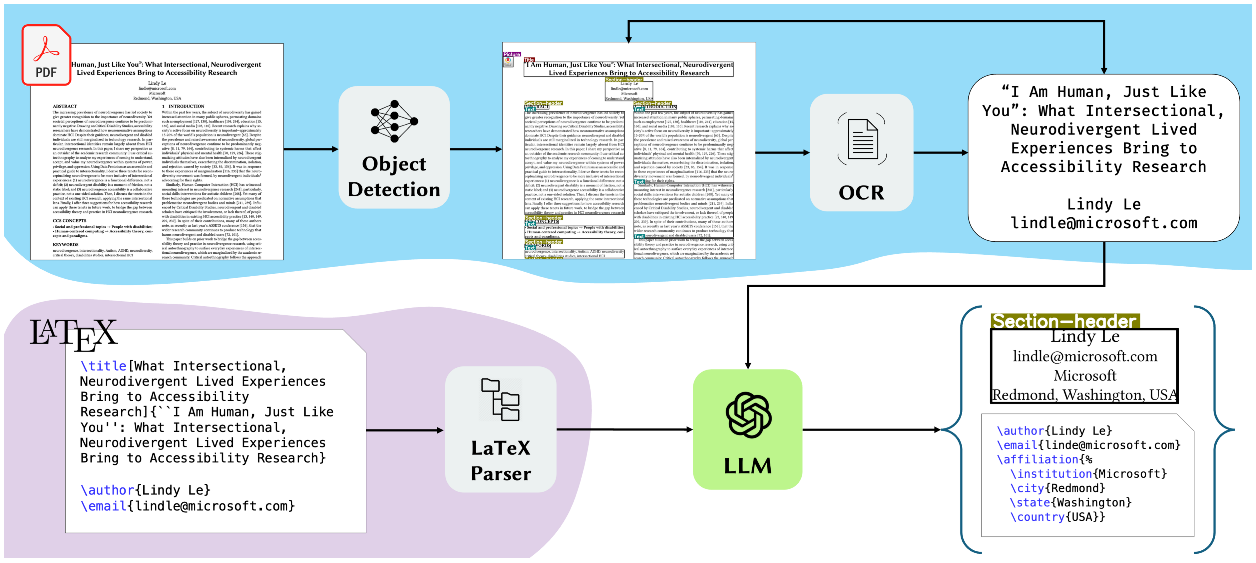
Consequently, tools such as SciA11y [39] and Paper to HTML [38] have been developed to convert PDFs into more accessible and interactive HTML formats. Complementarily, Brinn et *al.* [4] conducted user research suggesting that providing an HTML version of an academic paper, alongside the PDF and TeX source, could substantially improve the accessibility of the research posted on arXiv. Notably, Frankston et *al.* [8] observed that working directly from the source documents provided a superior starting point compared to PDFs, which is why platforms like ar5iv and the ACM Digital Library directly convert source files into accessible HTML. Despite these efforts, the uptake of alternative formats in scholarly communication remains severely limited, with even CHI and ASSETS venues requiring PDF submissions for review.

While prior work generally underscores the importance of source documents and the loss of semantic markup in visual PDFs, it is surprising that no research has addressed preserving this structure within the PDF itself. While seemingly straightforward, this problem remains unsolved, motivating us to explore whether recent advances in vision-language models can help bridge the gap.

2.3 Can Vision-Language Models Help?

Recent advances in document understanding rely heavily on machine learning pipelines (including large language models) that attempt to parse layout, extract text, and even answer questions (e.g., DocVQA [16]). These tasks are useful for information retrieval and limited forms of semantic linking, and they are sometimes used downstream in systems that convert PDFs to alternative formats (for example, both SciA11y and Paper to HTML use the GROBID ML pipeline [15]). However, these tasks remain narrow in scope and are unable to recover the precise spatial structure or order required to visually tag PDFs themselves.

Object Detection models, such as YOLO [31], trained or fine-tuned on document layout datasets (*e.g.,* PubLayNet [41], DocLayNet [28] or DocBank [13]) can often detect visual elements on pages like text and figures with high accuracy. However, they struggle to distinguish finer semantic categories like



```
“I Am Human, Just Like
You”: What Intersectional,
Neurodivergent Lived
Experiences Bring to
Accessibility Research

Lindy Le
lindle@microsoft.com
```

```
\title[What Intersectional,
Neurodivergent Lived Experiences
Bring to Accessibility
Research]{``I Am Human, Just Like
You'': What Intersectional,
Neurodivergent Lived Experiences
Bring to Accessibility Research}

\author{Lindy Le}
\email{lindle@microsoft.com}
```

Lindy Le
lindle@microsoft.com
Microsoft
Redmond, Washington, USA

```
\author{Lindy Le}
\email{linde@microsoft.com}
\affiliation{%
  \institution{Microsoft}
  \city{Redmond}
  \state{Washington}
  \country{USA}}
```

Figure 3: iTagPDF combines multiple representations of a PDF document. Illustrated in this figure are the steps it follows on a research paper PDF [12]: (i) processes the visual rendering of PDFs (blue background), (ii) parses the source document (lilac background) and finally, (iii) aligns the visual and source representations using a large language model (LLM).

paragraphs and captions. Moreover, they are limited to a fixed set of labels and often overlook elements important for accessibility, such as background artifacts. In theory, vision-language models (VLMs) like GPT-4V for object detection could overcome these limitations by using free-form text prompts to infer more nuanced structure. In practice, however, most existing approaches [5, 6, 14] perform poorly on document tagging tasks (see Figure 2), likely due to the scarcity of semantically tagged documents in their training sets.

Still, the notion of providing "context" offers a promising direction. Our key insight is that source documents can supply the missing semantic context needed for PDF tagging in these foundation models. In our work, we explore an in-context learning approach to build a semantically richer context by combining the source and visual representations of the PDF. Then, we build a system leverages this context to guide and refine visual tags in their corresponding PDFs.

3 Combining PDF Visual and Source Representations

We propose a new method that combines the visual and source representations of a PDF document to gather semantic context that can then be used further for tagging a PDF far more accurately than visual-only methods. Our method (see Figure 3) comprises of three steps: i) processing the visual rendering (PDF pages), ii) parsing the source document (*e.g.*, LaTeX files), and finally, iii) aligning both representations using a large language model (LLM).

3.1 Visual Rendering (PDF) Processing

The first step is to identify content regions and their respective types in page images extracted from PDFs. These regions may include headings, tables, figures, and paragraphs. This is achieved by performing inference on an object detection model fine-tuned on document images, which outputs bounding boxes, labels, and confidence scores for each PDF page.

Then, text contained on the page is extracted using an Optical Character Recognition (OCR) system. OCRs typically output word-level bounding boxes for all detected text on the page, along with the corresponding recognized text. Each word-level bounding box is then mapped to the corresponding predicted content region, and the recognized text is aggregated within each region. If textual content exists outside the current set of content regions, it is considered standalone and added as an individual `text` content region.

3.2 Source Document Parsing

Concurrently, the input source document(s) are linearized to form a single semantic context for further remediation of the generated content regions. This can be implemented via a custom LaTeX parser. Our parser processes the `main.tex` file and recursively incorporates content from all included files (e.g., `<section>.tex` and `main.bbl`).

This semantic context is pre-processed to remove comments, package imports, and other non-visual metadata. The cleaned document is then segmented into "chunks" – snippets of LaTeX code that are expected to render distinct visual content regions on a page. These chunks may include elements such as the title (`\title{}`), section headings (`\section{}` or

`\subsection{}`), authors (`\author{}`), lists `\begin{itemize}` or `\begin{enumerate}`), references (`@article{}`), footnotes (`\footnote{}`), and other meaningful constructs.

3.3 Aligning Visual and Source Representations

To combine visual and semantic representations, the OCR text output is aligned with the corresponding semantic chunk for each content region. Candidate semantic chunks are sorted and filtered based on their forward and reverse word overlap scores with the OCR content. Let T be the multi-set of words in the OCR text, and S be the multi-set of words in the semantic snippet, then these scores are computed as:

$$\text{Forward\_Overlap}(S,T) = \frac{|S \cap T|}{|T|} \tag{1}$$

$$\text{Reverse\_Overlap}(S,T) = \frac{|S \cap T|}{|S|} \tag{2}$$

Considering both these scores ensures the semantic chunk sufficiently explains the OCR text (forward), and that the OCR text meaningfully reflects the snippet chunk (reverse). This list of candidate chunks is then further filtered by prompting an LLM agent. Keeping an LLM in the loop helps reliably interpret noisy OCR text, recall semantic structure from ambiguous syntax, and handle edge cases that heuristics might struggle to resolve. The prompt instruction highlights given to the LLM agent are as follows:

Task: "You are given OCR text from a rendered LaTeX document and a list of LaTeX source chunks. Your task is to identify which specific LaTeX chunk(s) most likely rendered the visual content from the OCR output."
Input: OCR Text: `{ocr_text}`, LaTeX Chunks (with IDs): `[{chunk_snippet},...]`
Output: `JSON` ordered list of chunk identifiers, e.g., `["chunk_001","chunk_011",...]`
Instructions:

- Identify the smallest set of chunks that best explain the OCR text. Each selected chunk must contribute content not already covered by earlier ones. Do not include multiple chunks that contain the same content.
- It is acceptable for a chunk to contain extra content, as long as part of it matches the OCR. Matching does not need to be exact. (E.g., `\begin{abstract}`) may also render the ABSTRACT heading. However, the OCR text needs to reasonably be present in the code content.
- If the OCR text contains metadata likely from background elements (e.g., asides, repetitive headers and footers), such as the ACM reference format, submission dates, copyright information, or if none of the LaTeX chunks plausibly match the OCR text, return an empty list:`[]`

The LLM agent returns an initial mapping of semantic chunks to content regions, identifying which snippets most plausibly render each visual element. The content regions are further refined using this mapping through two targeted adjustments which are detailed below. These refinements are applied recursively until no further changes occur in the content regions or their associated mappings.

- **Splitting or Merging Content Regions**: When one content region maps to multiple semantic chunks, distinct elements (*e.g.*, a section and a subsection header) have been erroneously merged into one content region. Conversely, when multiple content regions correspond to a single semantic chunk, a coherent visual element, such as a paragraph or list, has been incorrectly fragmented into several bounding boxes. Bounding box area heuristics are used to correct such cases.
- **Mapping Figures to Semantic Chunks**: Since our mapping is based on text overlap, it does not associate visual elements with their corresponding semantic chunks (*i.e.*, `\begin{figure}` blocks). Content regions labeled as `Picture` in DocLayNet taxonomy are associated with the nearest text region that has already been assigned to a figure chunk (typically the caption). A visual proximity heuristic (threshold: 4% page height) is used to determine this mapping.

4 Intelligent PDF Tagging (iTagPDF)

iTagPDF processes the PDF visually to generate content regions and refines them by taking additional semantic context from the source documents. The specific implementations of the modules in iTagPDF are detailed in Table 1. Using a combined visual and source representation, iTagPDF ultimately produces and embeds content tags in a coherent reading order (see Figure 4), and other content specific accessibility metadata. After generating the accessibility metadata, iTagPDF automatically invokes a Java script using the iText Core library[6] to embed this metadata in the PDF.

4.1 Content Regions

After obtaining the mapped semantic chunk for each content region, iTagPDF applies a combination of regular expressions against the semantic chunk to correct or refine the visual label and generate appropriate content tags. A content region not mapped to any semantic chunk, *i.e.,* not added by the author in the source document, is treated as an `<Artifact>`. Our current implementation of iTagPDF produces 13 document-level tags: `<H1:H3>`, `<P>`, `<Caption>`, `<Table>`, `<L>`, `<Figure>`, `<Artifact>`, `<Note>`, `<Formula>`, `<BibEntry>`, and `<Author>`, and 3 content-specific tags: `<TH>`, `<TD>`, and `<LI>`.

[6]iText is a PDF SDK library for developers that provides enables low-level modification of the PDF structure, including the document's XMP metadata stream and the construction of a PDF tag tree.

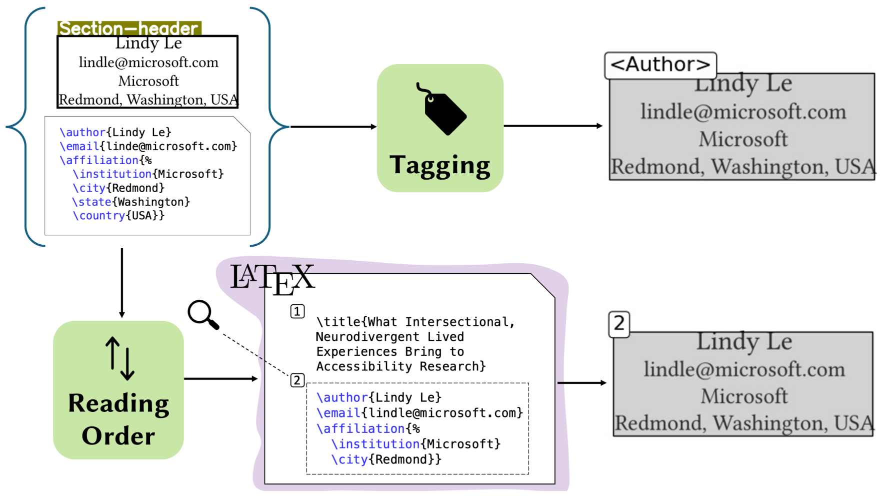


Figure 4: The combined mapping of visual and source representations are used to generate accessibility tags in a coherent reading order. The semantic chunk corresponding to a content region acts as an index for sorting the tags based on the original source documents.

4.2 Reading Order

iTagPDF considers the semantic chunk associated with each content region as an anchor for recovering the intended reading order of the PDF. Specifically, it sorts content regions on each page based on the position of their corresponding semantic chunk in the source LaTeX file. This ensures the reading order follows the author intent: for example, reading footnotes at their mention, paragraphs spanning columns uninterrupted, associating figures with their captions, and skipping artifacts.

4.3 Content Specific Metadata

4.3.1 Tables. iTagPDF further tags the structure of each table in the PDF. For each content region tagged as a `<Table>`, the region is cropped, and inference with another table-level object detector is performed on the cropped image. This object detector produces bounding boxes for each cell (wired or unwired) in the table. The text is extracted for each cell via the OCR module, and passed on to an LLM agent along with the semantic chunk associated with the table. The LLM agent is asked whether the cell is a row header, column header, or a data value. It's response is then parsed to further generate the `<TH>` and `<TD>` tags. The prompt instructions to the LLM are as follows:

Task: "You are given OCR text for a cell in the table and the LaTeX code that rendered the table. Your task is to identify whether the cell should be tagged as a table header (TH) or a table data (TD) cell. If the cell is a header, specify whether it is a row header or a column header."
Input:
Cell OCR Text: `{ocr_text}`
Table LaTeX Code: `{latex_code}`
Output:
Respond strictly in the format:
Label: TH or TD
Role: row or column or none (if not a header cell)

4.3.2 Headings. iTagPDF uses the semantic chunk for each heading to determine its level. The title (`\title{}`) of the paper is tagged as `<H1>`, while section (`\section{}`) and subsection headers (`\subsection{}`) are tagged as `<H2>` and `<H3>` respectively. iTagPDF also generates `<Author>` tags that map to the `<P>` role, following ACM's recommended practice for tagging author blocks in academic PDFs.

4.3.3 Lists. The current implementation of the document-level object detector in iTagPDF automatically identifies list items (`<LI>`) on the page. During content region refinement (Section 3.3), iTagPDF determines whether the list items correspond to a single semantic chunk, *i.e.*, a single list, and merges the corresponding regions to generate a list tag (`<L>`). Currently, iTagPDF does not handle references as list elements.

4.3.4 Formulas. Since the semantic chunks already contain the original LaTeX source for all mathematical equations in the PDF, iTagPDF embeds the raw LaTeX directly as its alt text, rather than attempting to re-render or translate the math. Our current implementation does not yet identify inline formulas, future versions of iTagPDF could include support for them.

4.3.5 Figures. For generating an alternative text description for a figure in the PDF, iTagPDF first checks if the author has already provided it in the LaTeX code (`\Description{}`). If this description is not found, iTagPDF prompts an LLM agent to automatically describe the figure. The prompt to the agent (derived from the instructions on creating accessible figures

Module	Implementation
Document-Level Object Detector	Ultralytics YOLOv11 model [9] fine-tuned[7] on the DocLayNet dataset [28]
Optical Character Recognition (OCR)	Google's Tesseract OCR Engine [34] via PyTesseract v0.3.13[8]
Parsing and Splitting LaTeX Files	Custom Python-based parser with regular expressions
Large Language Model (LLM) Agent	OpenAI GPT-4o
Table-Level Object Detector	Real-Time Detection Transformer (RT-DETR) [40] trained by PaddleOCR[9]
Embedding Generated Metadata in PDF	iText Core SDK/Library[10,11]

Table 1: Implementation of the different modules in the iTagPDF system.

provided to authors by ACM DIS[12]) is given in Appendix A.

5 Experiments

We performed a manual evaluation of iTagPDF by assembling a dataset of ACM research papers along with their source files. We adapted and redefined classical machine learning (ML) metrics to suit the specific challenges of PDF tagging, providing a more accurate estimate of automated tagging performance. Our system's results serve as a baseline for the broader research community. Finally, we manually assessed how well iTagPDF-generated metadata met the PDF accessibility criteria based on recent work [32].

5.1 Dataset Creation

Authors often post their pre-prints online on arXiv in the source document format. We searched for papers published at the ACM CHI and ASSETS conferences on arXiv over the past six years (2019–2024), filtering for entries that included source documents. We selected these conferences because ACM ASSETS requires authors to tag their PDFs for accessibility, while CHI strongly encourages it. Both venues are also well known for publishing accessible computing research, so authors are more likely to be familiar with accessibility standards. Additionally, we aimed to download the most recently published papers, assuming that authors would have access to the latest PDF remediation tools. As such, these PDFs were more likely to be tagged for accessibility. For each paper, we downloaded the LaTeX source from arXiv and the corresponding PDF from the ACM Digital Library to obtain the authors' final submitted, tagged versions. We discarded any entries where the arXiv version differed from the ACM version. In total, we collected 40 research paper PDFs (20 CHI; 20 ASSETS), comprising 554 pages (297 CHI; 257 ASSETS) and over 10,000 visual segments that required tagging.

[7]https://github.com/moured/YOLOv11-Document-Layout-Analysis
[8]https://pypi.org/project/pytesseract/
[9]https://www.paddleocr.ai/main/en/version3.x/module_usage/table_cells_detection.html
[10]https://itextpdf.com/solutions/universal-accessibility-pdfua
[11]https://github.com/itext/itext-java
[12]https://dis.acm.org/2023/creating-accessible-figures-and-tables/

5.2 Metadata Quality Metrics

To evaluate the quality of automatically generated tags and their reading order for accessibility, we redefined standard error rate metrics to better reflect the nuances of these tagging PDFs for accessibility. Traditional ML metrics for object detection, such as Intersection over Union (IoU) or mean Average Precision (mAP), are not directly aligned with the goals of content localization and semantic tagging for two reasons: *(i)* we lack precise ground truth annotations for these tasks, and *(ii)* there are multiple valid interpretations of how content may be tagged. For example, a region of text may be appropriately represented either as a single bounding box or as several smaller boxes, depending on semantic intent or tagging granularity. This also extends to reading order – multiple reading sequences may be equally valid depending on the content structure.

Instead, we focused on defining error metrics that can be computed manually:

(1) To determine the how well content on a page is localized, we define Box Error Rate analogous to Word Error Rate (WER):

$$BER = \frac{(I + D + S)}{N} \tag{3}$$

where I is the number of insertions (missed content with no bounding box), D is the number of deletions (extra or redundant boxes), S is the number of substitutions (bounding boxes that do not fully contain their intended content), and N is the total number of expected content regions. Assuming our method generates T tags, we compute the reference count as $N = T + I - D$. The values of I, D, and S were counted manually for each page. Then, bounding box accuracy score can be computed as Accuracy = 1 − BER.

(2) To evaluate tag classification, we define Classification Error Rate (CER) as:

$$\text{CER} = \frac{\text{Number of incorrectly classified tags (C)}}{\text{Total number of generated tags (T)}} \tag{4}$$

We count C by checking for classification mismatches manually. Redundant tags (D) should be labeled as artifact, else they are considered misclassified. The precision score of the predicted tag labels can then be computed as Precision = 1 − CER.

(3) Finally, we evaluate reading order by computing Reading Order Error Rate (ROER) as:

$$\text{ROER} = \frac{\text{Number of misplaced tags (M)}}{\text{Total number of generated tags (T)}} \quad (5)$$

If a tag does not follow the natural reading order, it is manually counted as misplaced. Similar to CER, we can compute reading order precision score as Precision = 1 − ROER.

5.3 Overall Accessibility Criteria

We further evaluated how well our generated metadata fulfills PDF accessibility criteria, as defined in a CHI 2025 paper [32]. While that work examined user performance with manual PDF remediation tools on tagging, specifying reading order and providing content specific metadata using these metrics, we evaluate iTagPDF, the first approach that fully automates PDF remediation against these criteria. We compare iTagPDF with two baselines: *(i)* Adobe Acrobat Pro Auto-Tagging, the most widely used PDF remediation software, and *(ii)* the higher standard of author-submitted PDFs to ACM CHI and ASSETS conferences [11]. Acrobat's cloud-based tagging feature, more accurate than its legacy on-device counterpart, was enabled to provide a stronger baseline for comparison.

6 Results

6.1 Metadata Quality

Overall, iTagPDF demonstrated high performance across all three tasks (content localization, semantic tagging and reading order identification) on our dataset, as shown in Table 2. It precisely distinguished between <Author>, <Note>, <Caption>, and <P> (Figure 5), which can sometimes be difficult even for authors to tag manually, as we will see in the next subsection.

Criteria	Error [%]	Score [%]
Bounding Boxes	4.72	95.28
Classification	4.03	95.96
Reading Order	2.74	97.26

Table 2: Quality of document-level tags, determined by manual evaluation of iTagPDF on our dataset ($N = 40$ PDFs). For further breakdown of bounding box error types, refer to Appendix B.

Visual elements, such as <Figure> and and <Table>, were most responsible for the incorrect bounding box errors in our results. Our refinements in Section 3.3 were not always able to correct content region boundaries, especially when dealing with non-textual content, such as figures or sub-figures (see Figure 6(b)). Future versions of iTagPDF could employ distinct object detector models to specifically identify these elements more accurately. Sometimes, elements were erroneously tagged as <Artifact>, making it the most redundant tag in our predictions (see Appendix B for more details). This could be attributed to either OCR being unable to process content in other languages (see Figure 6(a)), or generalization errors in our word overlap-based heuristics that failed to identify corresponding semantic chunks (most commonly for headings). We also observed a few instances where distinct content regions, *e.g.,* a caption and a table, or a header and the paper title, were combined or misclassified due to their semantic similarity.

6.2 PDF Accessibility Remediation

The metadata generated by iTagPDF on a subset ($N = 25$) of our dataset was much more accurate than Adobe Acrobat's auto-tagging approach on almost every criteria (depicted in Table 3). In fact, iTagPDF even surpassed authors' submitted manually tagged PDF metadata on several accessibility criteria: reading order, headings, lists, tables, and captions. Additionally, iTagPDF exhibited greater consistency across the evaluated papers compared to both baselines (see Appendix D for details).

Adobe Acrobat tends to automatically generate several empty <Span> tags, that inflate the number of tags generated on a page by approximately 10X magnitude. This makes manual evaluation of these tags extremely challenging and impractical. We found similar order of the number of tags in the author submitted PDFs as well, presumably because most authors use Acrobat to remediate their PDFs. Hence for evaluating these baselines, we counted the grouped tags displayed on the Acrobat user interface.

The reading order presented in this table was evaluated for the document-level tags. iTagPDF was successfully able to arrange paragraph spanning multiple columns together, even with figures or tables visually interrupting them. Further, iTagPDF was able to tag author names and affiliations in the correct order, whereas other baselines often failed to do so. Unlike Acrobat, iTagPDF correctly tagged repetitive headers as <Artifact> and excluded them from the reading order.

We evaluated table structures for a smaller subset of our dataset ($N = 5$) since a complete evaluation was not feasible given their high frequency. Adobe Acrobat performed tagged most tables highly accurately in our dataset. Since this result is inconsistent with prior work [32], we suspect Adobe has improved their backend model powering structured tagging for tables. However, while their model is not publicly accessible to the academic community, our work is the first open and reproducible method for tagging table structures in PDFs. Errors in table structure tagging for iTagPDF included cases with ambiguity in interpreting the first entry of each row as either a row header or a data value. Because our current implementation of iTagPDF prompts an LLM for each cell individually, the resulting outputs can sometimes be inconsistent. Future work on iTagPDF may include asking an LLM higher-level

	Adobe Acrobat Auto-Tags	**What Authors Submitted**			**iTagPDF (Ours)**
Criteria	[%]	**CHI** [%]	**ASSETS** [%]	**Total** [%]	[%] (N)
All Content Tagged	100.0 (6167)	100.0 (1787)	100.0 (1584)	100.0 (3371)	100.0 (5087)
Reading Order	85.53	84.44	93.37	88.62	**96.42**
Headings Tagged	65.31	39.70	95.36	71.92	91.58 (618)
+ Level	62.26	39.70	**95.09**	71.76	89.79
Tables Tagged	73.91	81.25	59.37	66.67	89.79 (49)
+ Structure (N = 5)	**81.29**	72.20	42.85	62.59	72.40 (877)
Lists Tagged	58.22	11.76	83.33	46.46	83.09 (71)
+ Structure	61.44	49.36	60.00	51.51	**92.30** (195)
Figures Tagged	57.93	76.90	84.93	81.60	75.20 (121)
+ Alt Text	0.0	76.90	84.93	**81.60**	75.20
Formulas Tagged	0.0	0.0	–	0.0	50.00 (2)
+ Alt Text	0.0	0.0	–	0.0	**50.00**
Captions Tagged	0.0	0.0	0.0	0.0	**82.92** (164)
Average Score	49.68	48.63	72.65	55.59	**81.36**
Normalized Score	87.03	79.52	81.53	80.97	**95.19**

Table 3: Comparison of PDF accessibility tagging between metadata generated by Acrobat's auto-tagging feature, what authors submitted and iTagPDF. To compare the results, we manually evaluated 25 randomly selected papers from our dataset. The parenthesis indicates the total number of elements evaluated against the criteria.

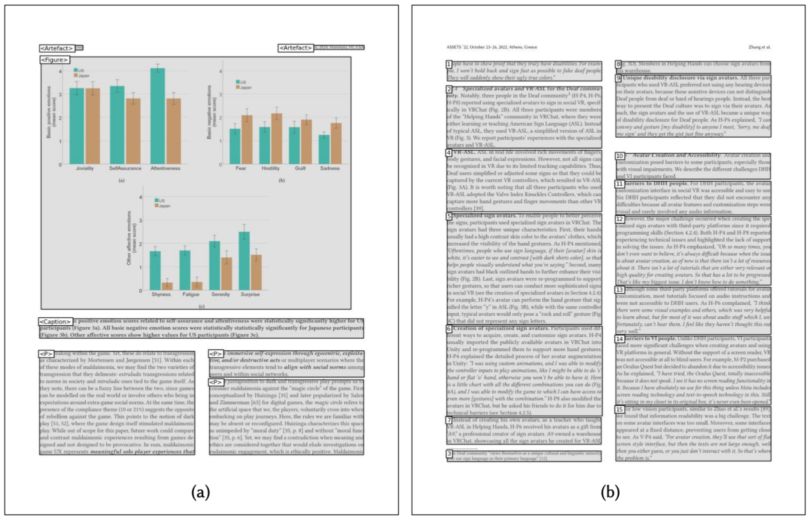


Figure 5: Positive examples of iTagPDF-generated metadata: (a) Figures, captions, and artifacts are accurately classified; (b) Footnote order is correctly predicted, repetitive headers are skipped.

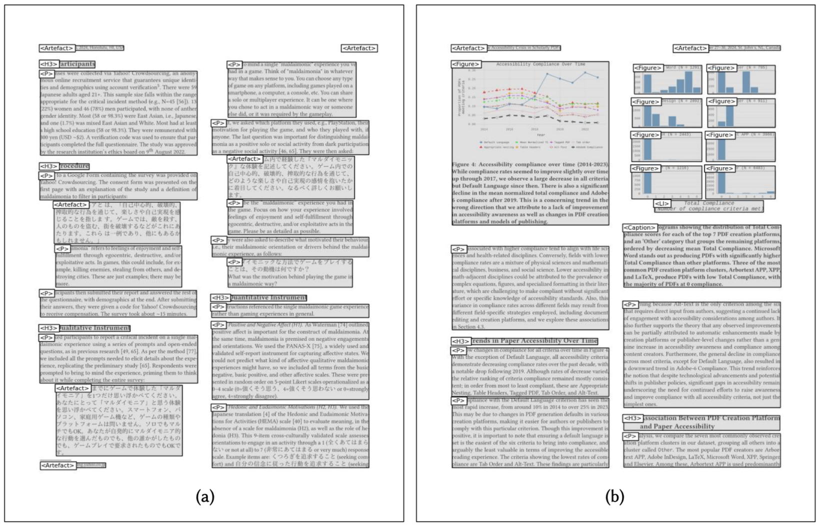


Figure 6: Negative examples of iTagPDF-generated metadata: (a) Paragraphs in Japanese are tagged artifacts; (b) Figures are occasionally merged with captions or split into sub-figures.

questions such as whether a table contained any row or column headers. Interestingly, by leveraging semantic context from the LaTeX table syntax, iTagPDF was occasionally able to identify headers spanning multiple rows or columns. In contrast, Adobe Acrobat assumed only the first row or column to serve as a header in such cases.

Overall, all the baselines were reasonably accurate at identifying figures on a page. However, Adobe Acrobat at times mislabeled artifacts, tables, and non-English text as figures, which contributed to its low score. Every figure in the PDFs authors submitted had an alternative text description provided by the authors. While iTagPDF also generates alt text for figures, evaluating the quality of alt text generated is beyond the current scope of this work. Researchers are actively studying interactive LLM-guided tooling for producing alt text [33] and future work could also look at evaluation of such automatic and semi-automatic alt text generation approaches. Interestingly, neither the authors nor Adobe tagged any captions accurately in our dataset. Currently, Acrobat cannot automatically distinguish between captions and paragraphs, perhaps due to the absence of caption labels in its training data.

Our evaluation aligns with prior accessibility assessments of ASSETS and CHI papers [3], showing that papers submitted to ASSETS are significantly more compliant with accessibility standards than those submitted to CHI. Moreover, iTagPDF outperforms CHI authors at remediating PDFs for accessibility on nearly all criteria, indicating that future CHI conference proceedings could potentially benefit from automated semantic tagging methods.

7 Discussion

This paper presented a novel method for preserving the semantics of source documents to automatically enhance PDF accessibility. Reflecting on our results, we discuss how shifting attention from metadata remediation to preservation can effectively resolve many stubborn challenges in automating PDF accessibility. Moreover, as automation grows increasingly powerful, we consider the promise and pitfalls of reducing, or entirely eliminating, human intervention in accessibility enhancement workflows.

7.1 Limitations & Future Work

We identify several limitations in the current implementation of iTagPDF, and outline promising directions for future work.

Firstly, our work focuses on research paper PDFs authored in LaTeX. Despite being a fairly popular academic authoring tool, LaTeX-borne PDFs often struggle with accessibility compliance [10]. While we demonstrate our method using LaTeX source files, other types of source documents, such as Microsoft Word or Adobe InDesign files, may also contain valuable semantic information that can be lost during PDF rendering. Extending our approach to these formats is a natural next step for this work. In addition, although our evaluation focuses on scientific publications from ACM venues, we empirically observed that our approach generalizes well to other types of publishers (See Appendix E). Future work should investigate how well the our method performs in other contexts where PDF accessibility remains challenging, including newspapers [22], government and digital forms [25, 35], and a wide range of technical, leisure and commerce reports.

Secondly, iTagPDF currently relies on a combination of heuristics and LLM-in-the-loop to map the visual and source representations of the PDF document and resolve any conflicts between them. This approach is rigid and can fail when a reliable mapping is not found (e.g., for figures or captions), resulting in redundant <Artifact> tags). A promising future direction is constructing larger training datasets that pair the visual rendering of a PDF with its underlying source representation. These datasets would support combined training over both semantic and visual representations, improving robustness, generalization, and the system's ability to infer semantics even when the source representation is missing. Since iTagPDF leverages an LLM, its outputs are not strictly deterministic. To assess system stability, we conducted an empirical evaluation (detailed in Appendix C) and found that it is nearly deterministic for most tags. Further opportunities include exploring better LLM architectures for improving consistency in tagging table elements and reducing the overall pipeline latency.

Third, iTagPDF's accuracy is limited by the performance of its underlying off-the-shelf modules. For example, our page-level object detector sometimes struggles to distinguish between figures, sub-figures and their captions. We conducted qualitative comparisons across state-of-the-art models to determine each component (e.g., see Figure 2 for comparison among different object detectors). As these models continue to advance, we also expect corresponding improvements in iTagPDF's performance.

Finally, while iTagPDF provides a strong baseline for automated PDF tagging, it does not produce some tags (such as `<Link>`), and still relies on author verification to preserve authoring agency. For instance, authors must manually review and refine the generated figure alt-texts to ensure essential information is conveyed accurately. While we primarily envision iTagPDF as an authoring or publishing tool, end-users might also employ it to remediate poorly tagged PDFs, provided that the corresponding source files are available.

7.2 From PDF Accessibility "Remediation" to "Preservation"

Most accessibility research aims to make content accessible in one of two ways: i) enabling authors to create accessible content from the outset (*i.e.,* borne-accessible), and ii) retrofitting accessibility into existing content (*i.e.,* remediation). Because of various inconsistencies in how PDF creation tools render files, and also the complexity of the format itself, most work on PDF accessibility is pigeonholed into remediation. Remediating PDFs based on their visual rendering raises ambiguities that have frustrated researchers for more than a decade. For example, making math accessible often begins with conversion of the visual formula to an unambiguous format such as TEX or MathML [24]. However, challenges remain in disambiguating symbols and mapping their intended meaning [20]. With the increasing capabilities of vision-language models (VLMs), one might expect them to leverage their world knowledge to provide the required context for disambiguating such cases. However, VLMs too struggle with building deeper associations as they are trained for narrow tasks on image-text pairs scraped from the web [37].

The middle ground between borne-accessible content and post-hoc remediation lies in the preservation of important semantic data that is present at creation time but often gets lost during rendering. Pareddy *et al.* [26] first proposed this idea for embedding user interface structural information into screenshot images. They explained that keeping this information at capture time would help screen reader users navigate the screenshots in line with the original designer's intent.

Similarly, our tagging method focused on retrieving semantic context from source documents for PDF remediation. We employed a retrieval-augmented generation (RAG) approach to incorporate relevant context for resolving visual ambiguities in tagging PDFs. Once this "ground truth" context was available, we could directly generate the required tags, without any additional training, even though these tags differed from the labels used by the original object detection models (see Table 4). Now, rather than converting equations from their visual rendering to LaTeX, we could directly access the LaTeX as originally authored. Our results show that preservation of this semantic metadata from the source can enable powerful automation in accessibility remediation.

Original Training Labels	**iTagPDF Tag(s)**
DocLayNet Labels	
title	<H1>
section-header	<H2>, <H3>
picture	<Figure>
text	<P>, <Author>, <BibEntry>
list-item	<L>, <LI>
table	<Table>
caption	<Caption>
footnote	<Note>
formula	<Formula>
page-header, page-footer	<Artifact>
Table-Cell-Detection Labels	
cell	<TH>, <TD>

Table 4: Original training labels and corresponding iTagPDF tags, organized by document-level and table-level elements.

7.3 The Promise and Pitfalls of Automating PDF Accessibility

While AI-driven accessibility as a full solution was once anathema to the accessibility community, even some long-term accessibility advocates are starting to wonder when (and if) automated approaches might eventually replace author accessibility requirements as it becomes increasingly performant [36], although none would recommend fully relying on this approach now. A question going forward will be how and whether to support a responsible transition toward more and more automation.

In this specific case of PDF accessibility, authors have been considered the primary stakeholders responsible for making their documents accessible. They have long complained about the tedious process of PDF tagging, the difficulty of applying accessibility practices to complex visual elements, the poor usability of existing tools, and even knowing whether they are doing it correctly [2, 17]. Automating these aspects of PDF accessibility certainly alleviates the burden on the authors. However, authors still need to be aware and accepting of such tools to use them. Other secondary stakeholders, such as publishers, might also be able to run a semantic tagging system such as iTagPDF that requires only the source documents and the visual rendering.

However, even "silent" tools that automate accessibility with minimal human intervention still require user oversight and understanding to perform effectively [23]. A tension in this work is that automating PDF remediation entirely could risk making authors complacent in supplying necessary context manually. For instance, authors must still provide or validate alternative text descriptions for figures. Further, over-reliance on automated PDF accessibility tooling might inadvertently reinforce the continued dominance of PDFs in academic publishing over formats like HTML and ePub which offer far better accessibility support. However, given the pervasiveness of the PDF format in current research dissemination practices, we believe improving accessibility through better authoring and PDF remediation tools remains necessary.

8 Conclusion

In this paper, we introduce a novel approach that combines visual tagging of PDFs with semantic cues from the source document. We develop iTagPDF that applies this approach to generate more accurate and robust PDF accessibility metadata. iTagPDF offers what we believe to be the most reliable solution to date for automated PDF remediation and surpasses the quality of manual tagging provided by authors in our collected dataset. It dramatically lowers author effort and removes usability barriers present in the current process of making PDFs accessible. Our work represents a concrete first step toward the long-standing goal of accurately and automatic PDF accessibility remediation.

References

[1] Adobe Inc. 2023. *Adobe Acrobat Pro DC Accessibility Features.* Adobe Systems Incorporated. https://helpx.adobe.com/acrobat/using/create-verify-pdf-accessibility.html.

[2] Jeffrey P. Bigham, Erin L. Brady, Cole Gleason, Anhong Guo, and David A. Shamma. 2016. An Uninteresting Tour Through Why Our Research Papers Aren't Accessible. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 621–631. doi:10.1145/2851581.2892588

[3] Erin Brady, Yu Zhong, and Jeffrey P. Bigham. 2015. Creating accessible PDFs for conference proceedings. In *Proceedings of the 12th International Web for All Conference* (Florence, Italy) *(W4A '15)*. Association for Computing Machinery, New York, NY, USA, Article 34, 4 pages. doi:10.1145/2745555.2746665

[4] Shamsi Brinn, Christopher Cameron, David Fielding, Charles Frankston, Alison Fromme, Peter Huang, Mark Nazzaro, Stephanie Orphan, Steinn Sigurdsson, Ryan Tay, Miranda Yang, and Qianyu Zhou. 2024. A framework for improving the accessibility of research papers on arXiv.org. arXiv:2212.07286 [cs.DL] https://arxiv.org/abs/2212.07286

[5] Jiaqi Chen, Zeyu Yang, and Li Zhang. 2023. Semantic Segment Anything. https://github.com/fudan-zvg/Semantic-Segment-Anything.

[6] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. 2024. YOLO-World: Real-Time Open-Vocabulary Object Detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 16901–16911. doi:10.1109/CVPR52733.2024.01599

[7] Equidox Software Company. 2025. Equidox PDF Accessibility Software. https://equidox.co/pdf-solutions/pdf-accessibility-software/. Accessed: 2025-04-17.

[8] Charles Frankston, Jonathan Godfrey, Shamsi Brinn, Alison Hofer, and Mark Nazzaro. 2024. HTML papers on arXiv – why it is important, and how we made it happen. arXiv:2402.08954 [cs.DL] https://arxiv.org/abs/2402.08954

[9] Glenn Jocher and Jing Qiu. 2024. *Ultralytics YOLO11*. https://github.com/ultralytics/ultralytics

[10] Anukriti Kumar and Lucy Lu Wang. 2024. Uncovering the New Accessibility Crisis in Scholarly PDFs: Publishing Model and Platform Changes Contribute to Declining Scholarly Document Accessibility in the Last Decade. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) *(ASSETS '24)*. Association for Computing Machinery, New York, NY, USA, Article 36, 16 pages. doi:10.1145/3663548.3675634

[11] Jonathan Lazar, Elizabeth F. Churchill, Tovi Grossman, Gerrit van der Veer, Philippe Palanque, John "Scooter" Morris, and Jennifer Mankoff. 2017. Making the field of computing more inclusive. *Commun. ACM* 60, 3 (Feb. 2017), 50–59. doi:10.1145/2993420

[12] Lindy Le. 2024. "I Am Human, Just Like You": What Intersectional, Neurodivergent Lived Experiences Bring to Accessibility Research. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) *(ASSETS '24)*. Association for Computing Machinery, New York, NY, USA, Article 51, 20 pages. doi:10.1145/3663548.3675651

[13] Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. 2020. DocBank: A Benchmark Dataset for Document Layout Analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, Donia Scott, Nuria Bel, and Chengqing Zong (Eds.). International Committee on Computational Linguistics, Barcelona, Spain (Online), 949–960. doi:10.18653/v1/2020.coling-main.82

[14] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. 2025. Grounding DINO: Marrying DINO with Grounded Pre-training for Open-Set Object Detection. In *Computer Vision – ECCV 2024*, Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (Eds.). Springer Nature Switzerland, Cham, 38–55.

[15] Patrice Lopez. 2009. GROBID: Combining Automatic Bibliographic Data Recognition and Term Extraction for Scholarship Publications. In *Research and Advanced Technology for Digital Libraries*, Maristella Agosti, José Borbinha, Sarantos Kapidakis, Christos Papatheodorou, and Giannis Tsakonas (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 473–474.

[16] Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. 2021. DocVQA: A Dataset for VQA on Document Images. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2199–2208. doi:10.1109/WACV48630.2021.00225

[17] Rachel Menzies, Garreth W. Tigwell, and Michael Crabb. 2022. Author Reflections on Creating Accessible Academic Papers. *ACM Trans. Access. Comput.* 15, 4, Article 33 (Oct. 2022), 36 pages. doi:10.1145/3546195

[18] Frank Mittelbach and Ulrike Fischer. 2024. Enhancing LATEX to automatically produce tagged and accessible PDF. *TUGboat* 45, 1 (2024), 52–59. doi:10.47397/tb/45-1/tb139mitt-deims24

[19] Frank Mittelbach, Ulrike Fischer, David Carlisle, and Joseph Wright. 2024. Automatically producing accessible and reusable PDFs with LATEX. In *Proceedings of the ACM Symposium on Document Engineering 2024* (San Jose, CA, USA) *(DocEng '24)*. Association for Computing Machinery, New York, NY, USA, Article 15, 4 pages. doi:10.1145/3685650.3685670

[20] Frank Mittelbach, Ulrike Fischer, David Carlisle, and Joseph Wright. 2025. MathML and other XML Technologies for Accessible PDF from LATEX. In *Proceedings of the 2025 ACM Symposium on Document Engineering* (Nottingham, United Kingdom) *(DocEng '25)*. Association for Computing Machinery, New York, NY, USA, Article 19, 4 pages. doi:10.1145/3704268.3748669

[21] Frank Mittelbach and Chris Rowley. 2020. LATEX Tagged PDF—A blueprint for a large project. *TUGboat* 41, 3 (2020), 292–298. doi:10.47397/tb/41-3/tb129mitt-tagpdf

[22] Peya Mowar, Meghna Gupta, and Mohit Jain. 2024. Breaking the News Barrier: Towards Understanding News Consumption Practices among BVI Individuals in India. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) *(ASSETS '24)*. Association for Computing Machinery, New York, NY, USA, Article 66, 11 pages. doi:10.1145/3663548.3675608

[23] Peya Mowar, Yi-Hao Peng, Jason Wu, Aaron Steinfeld, and Jeffrey P Bigham. 2025. CodeA11y: Making AI Coding Assistants Useful for Accessible Web Development. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 45, 15 pages. doi:10.1145/3706598.3713335

[24] Azadeh Nazemi, Iain Murray, and Nazanin Mohammadi. 2012. Mathspeak: An Audio Method for Presenting Mathematical Formulae to Blind Students. In *2012 5th International Conference on Human System Interactions*. 48–52. doi:10.1109/HSI.2012.17

[25] Sparsh Paliwal, Joshua Hoeflich, J. Bern Jordan, Rajiv Jain, Vlad I. Morariu, Alexa Siu, and Jonathan Lazar. 2025. FormA11y—Research and Development of a Tool for Remediating PDF Forms for Accessibility. *ACM Trans. Comput.-Hum. Interact.* 32, 1, Article 5 (April 2025), 39 pages. doi:10.1145/3702317

[26] Sujeath Pareddy, Anhong Guo, and Jeffrey P. Bigham. 2019. X-Ray: Screenshot Accessibility via Embedded Metadata. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility* (Pittsburgh, PA, USA) *(ASSETS '19)*. Association for Computing Machinery, New York, NY, USA, 389–395. doi:10.1145/3308561.3353808

[27] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. Association for Computing Machinery, New York, NY, USA, Article 2, 22 pages. doi:10.1145/3586183.3606763

[28] Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar. 2022. DocLayNet: A Large Human-Annotated Dataset for Document-Layout Segmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 3743–3751. doi:10.1145/3534678.3539043

[29] Debashish Pradhan, Tripti Rajput, Aravind Jembu Rajkumar, Jonathan Lazar, Rajiv Jain, Vlad I. Morariu, and Varun Manjunatha. 2022. Development and Evaluation of a Tool for Assisting Content Creators in Making PDF Files More Accessible. *ACM Trans. Access. Comput.* 15, 1, Article 3 (March 2022), 52 pages. doi:10.1145/3507661

[30] Aravind Jembu Rajkumar, Jonathan Lazar, J Bern Jordan, Alireza Darvishy, and Hans-Peter Hutter. 2020. PDF accessibility of research papers: What tools are needed for assessment and remediation?. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*. 1–10. doi:10.24251/HICSS.2020.512

[31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 779–788. doi:10.1109/CVPR.2016.91

[32] Felix Maximilian Schmitt-Koopmann, Elaine May Huang, Hans-Peter Hutter, and Alireza Darvishy. 2025. Towards More Accessible Scientific PDFs for People with Visual Impairments: Step-by-Step PDF Remediation to Improve Tag Accuracy. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 48, 16 pages. doi:10.1145/3706598.3713084

[33] Nikhil Singh, Lucy Lu Wang, and Jonathan Bragg. 2024. FigurA11y: AI Assistance for Writing Scientific Alt Text. In *Proceedings of the 29th International Conference on Intelligent User Interfaces* (Greenville, SC, USA) *(IUI '24)*. Association for Computing Machinery, New York, NY, USA, 886–906. doi:10.1145/3640543.3645212

[34] R. Smith. 2007. An Overview of the Tesseract OCR Engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, Vol. 2. 629–633. doi:10.1109/ICDAR.2007.4376991

[35] Utku Uckun, Ali Selman Aydin, Vikas Ashok, and IV Ramakrishnan. 2020. Breaking the Accessibility Barrier in Non-Visual Interaction with PDF Forms. *Proc. ACM Hum.-Comput. Interact.* 4, EICS, Article 80 (June 2020), 16 pages. doi:10.1145/3397868

[36] Gregg Vanderheiden and Crystal Yvette Marte. 2024. Will AI allow us to dispense with all or most accessibility regulations?. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '24)*. Association for Computing Machinery, New York, NY, USA, Article 571, 9 pages. doi:10.1145/3613905.3644059

[37] Lucy Lu Wang. 2024. Generative AI for Scholarly Information Access. *Charleston Hub's Against the Grain* 36, 3 (June 2024). https://issuu.com/against-the-grain/docs/june_2024_v36-3/s/52583209

[38] Lucy Lu Wang, Jonathan Bragg, and Daniel S. Weld. 2023. Paper to HTML: A Publicly Available Web Tool for Converting Scientific Pdfs into Accessible HTML. *SIGACCESS Access. Comput.* 134, Article 1 (Jan. 2023), 1 pages. doi:10.1145/3582298.3582299

[39] Lucy Lu Wang, Isabel Cachola, Jonathan Bragg, Evie Yu-Yen Cheng, Chelsea Haupt, Matt Latzke, Bailey Kuehl, Madeleine van Zuylen, Linda Wagner, and Daniel S. Weld. 2021. Improving the Accessibility of Scientific Documents: Current State, User Needs, and a System Solution to Enhance Scientific PDF Accessibility for Blind and Low Vision Users. arXiv:2105.00076 [cs.DL] https://arxiv.org/abs/2105.00076

[40] Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. 2024. DETRs Beat YOLOs on Real-time Object Detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 16965–16974. doi:10.1109/CVPR52733.2024.01605

[41] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. PubLayNet: Largest Dataset Ever for Document Layout Analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 1015–1022. doi:10.1109/ICDAR.2019.00166

A Generating Figure Alt-Texts

The prompt to the LLM agent for producing an alternative text description for figures in the PDF is as follows:

Task: "Your task is to write an alternative text description for a scientific figure. Alt text makes visual content (e.g., images, diagrams, and charts) accessible to screen reader and braille users. They include the essential information about a figure."
Input: `{figure}`
Output: Only the alt text.
Instructions:

- Write alt text considering the context and audience. While many guides recommend that alt text be brief, remember the audience for papers and pictorials is often other researchers. Consider what an alt text reader would need to know to adequately review whether the work is relevant for their own research–such as accurately summarizing your results in their own literature review or even replicating the study. This may require more detail than alt text written for a more casual context like social media.
- Generally, structure your alt text to give a high-level summary, then details. However, high-level summaries should not be the same as the figure caption, which a screen reader user will also read. Whereas the caption will provide context (guidance in the previous section), the high-level overview can overview the visual content (e.g., 2 line charts (top and bottom) depicting changes over time). The details can then commence. (E.g., (top) line chart titled T shows A on the X-axis, B on the Y-axis and contains C trend and/or D and E key data points.) Sharing key takeaways that the visual draws a reader's eye toward may be included in this detail, particularly when a graphic is complex with many data series, datapoints, or trend changes.
- Describe relevant research information. When depicting study equipment, consider its high importance to the figure. A lot of alt text guidance emphasizes

the importance of describing people, sometimes more than other objects in the image. However, in the context of a paper or pictorial, alt text readers are likely also researchers and need to know more details about the study equipment, prototype, workshop supplies, etc.

- If you are writing alt text for a data representation, it is likely essential that you include the chart/diagram type, the type of data in the chart which is often reflected in the axis labels and legend, and the key takeaways. Colors, shapes, and textures used to visually style a data representation do not need to be described, but the categories they represent should be included.
- Some figures may show meaningful relationships, such as arrows between boxes in a diagram or schematic. Explain meaningful relationships in alt text after giving the high-level overview.
- Some figures may depict a group of images. If the grouping is meaningful, write this meaning in the alt text before describing the individual images.

B Breakdown of Metadata Errors

Table 5 lists further breakdown of the bounding box errors (4.72%). Figure 7 illustrates the frequency of tag classification errors.

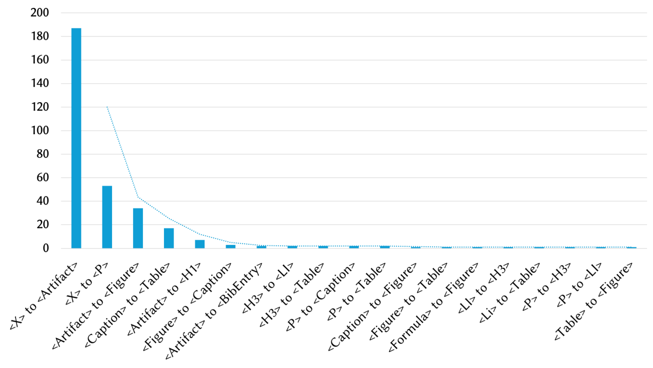


Figure 7: Classification error frequencies across 40 PDFs comprising 554 pages ($N = 10,261$ tags). Each bar represents the frequency of misclassification from a ground truth tag to a predicted tag (denoted as `<True>` to `<Predicted>`). `<X>` denotes all the tags in our dataset. The most common error involves classifying various tags as `<Artifact>`, indicated by `<X>` to `<Artifact>`.

C Tagging Stability

We evaluated iTagPDF on $N = 5$ randomly selected PDFs from our dataset, performing $i = 3$ independent runs per document. Table 6 reports the aggregate averages and standard deviations for different accessibility tags. Overall, iTagPDF demonstrates consistent performance across runs, with a normalized accuracy of 94.06% and a low variance of $\sigma = 0.80$. While its performance is highly stable for reading order and most tags, it shows greater variation for the `<Table>` and `<Caption>` tags.

Error Types	**Distribution [%]**
Insertions	38.77
Deletions	26.06
Substitutions	35.17

Table 5: Distribution of bounding box errors by iTagPDF on our dataset ($N = 40$ PDFs).

Criteria	**Accuracy** [μ %] (σ)
All Content Tagged	100.0 (0.0)
Reading Order	94.66 **(0.49)**
Headings Tagged	79.93 (3.23)
+ Level	78.25 (3.20)
Tables Tagged	77.58 (10.85)
Lists Tagged	83.00 (0.57)
Figures Tagged	77.08 (1.80)
Formulas Tagged	100.0 (0.0)
Captions Tagged	54.84 (15.18)
Normalized Average	94.06 **(0.80)**

Table 6: Average accuracies (μ %) and standard deviations (σ) calculated for $N = 5$ random PDFs on our dataset across $i = 3$ runs of iTagPDF.

D Tagging Consistency

Since users experience accessibility at the document level, we aggregated our results (in Table 3) per paper to evaluate the variation in iTagPDF's performance across documents in our dataset. The detailed results are provided in Table 7. Overall, iTagPDF not only achieves higher tagging accuracy per paper but also exhibits lower performance variability across papers compared to other baselines for most accessibility criteria. This suggests that, while purely visual tagging methods are highly sensitive to document content and layout complexity, semantic tagging provides a more consistent and reliable approach.

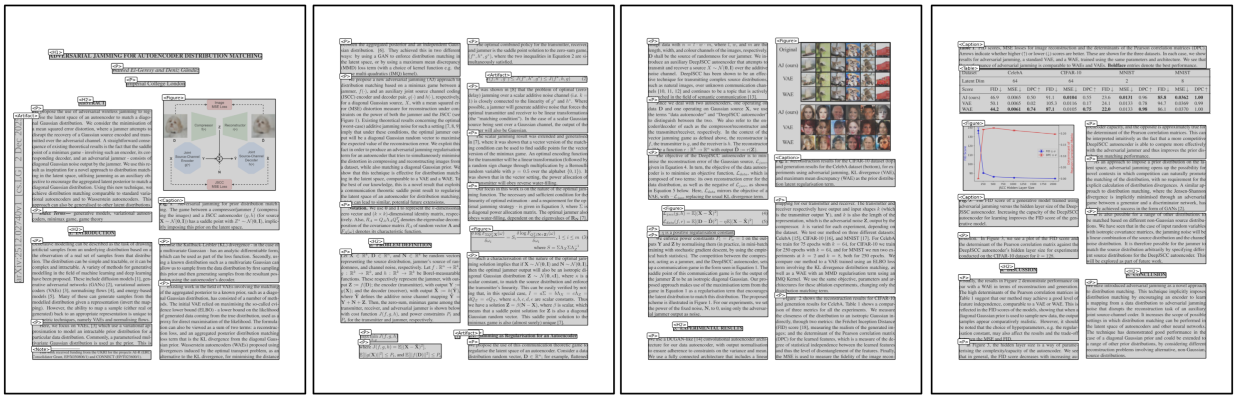
E Tagging Generalization

We applied iTagPDF without modifications to academic papers from diverse publishers and domains, spanning audio processing, machine learning, and physics. The qualitative results are shown in Figure 8. Overall, iTagPDF generalizes reasonably

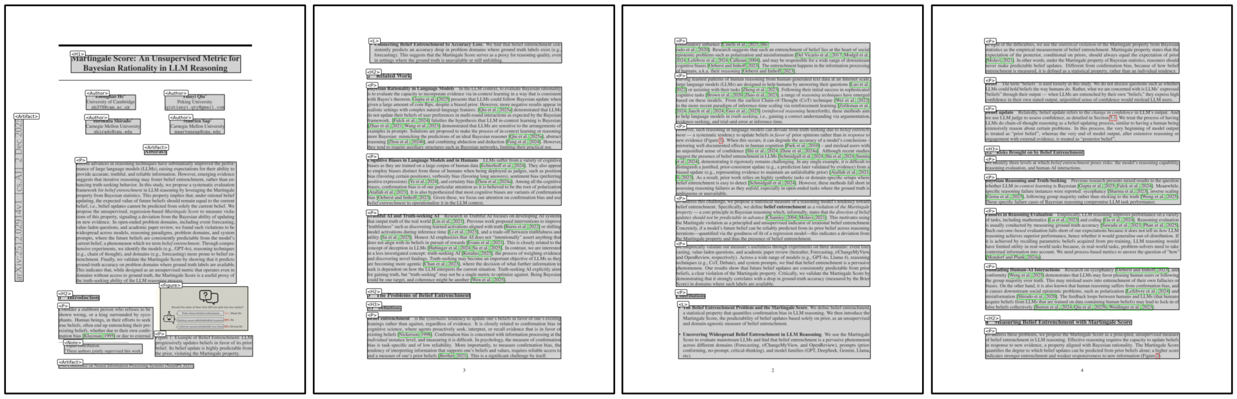
Criteria	**Adobe Acrobat Auto-Tags** [%]	**What Authors Submitted** [%]	**iTagPDF (Ours)** [μ %] (σ, N)
All Content Tagged	100.0 (0.0, 247)	100.0 (0.0, 135)	100.0 (0.0, 203)
Reading Order	77.24 (16.05)	86.62 (19.41)	96.31 (**2.31**, 203)
Headings Tagged	62.3 (45.36)	74.14 (42.09)	91.59 (**12.35**, 25)
+ Level	58.79 (46.65)	73.98 (42.00)	91.53 (13.48, 25)
Tables Tagged	73.68 (42.06)	75.52 (33.74)	86.55 (**28.22**, 2)
+ Structure (N = 5)	86.61 (29.93)	62.69 (29.87)	74.30 (25.15, 175)
Lists Tagged	61.15 (42.75)	72.92 (**13.26**)	86.95 (27.54, 3)
+ Structure	64.29 (57.52)	73.06 (39.45)	92.10 (12.76, 8)
Figures Tagged	64.29 (57.52)	83.77 (43.04)	79.20 (**20.32**, 5)
+ Alt Text	0.0 (0.0)	88.31 (38.84)	79.20 (20.32, 5)
Formulas Tagged	0.0 (0.0)	0.0 (0.0)	50.00 (70.71, 0)
+ Alt Text	0.0 (0.0)	0.0 (0.0)	50.00 (70.71, 0)
Captions Tagged	0.0 (0.0)	0.62 (2.98)	86.71 (20.58, 7)
Normalized Average	77.93 (19.71)	83.97 (15.46)	94.99 (**4.37**)

Table 7: Comparison of PDF accessibility tagging between metadata generated by Acrobat's auto-tagging feature, what authors submitted and iTagPDF on 25 randomly selected papers from our dataset, aggregated at the paper level. Here, μ denotes the average score, σ denotes the standard deviation and N denotes the average number of elements per paper rounded off to the nearest integer.

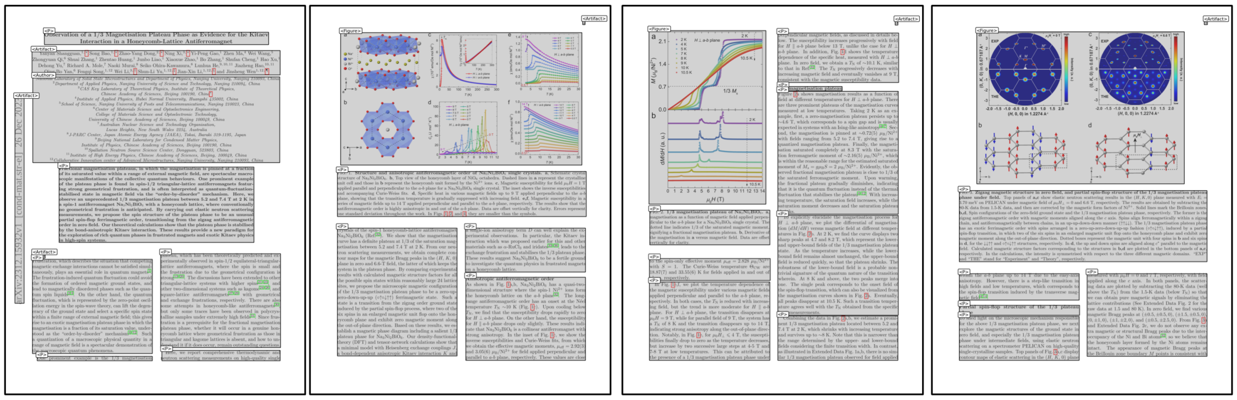
well to these documents, though it occasionally struggles to identify the `<Author>`, `<Caption>`, and `<Formula>` tags.



(a) IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)



(b) The Annual Conference on Neural Information Processing Systems (NeurIPS)



(c) Nature Physics

Figure 8: Qualitative examples of iTagPDF's tagging performance on academic papers across diverse publication venues and document templates: (a) IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (b) The Annual Conference on Neural Information Processing Systems (NeurIPS), and (c) Nature Physics.