# **DECODING PDFS: TRANSFORMING DOCUMENTS INTO MARKDOWN**

# **PROBLEM STATEMENT**

# Organizations and researchers often need to extract and process data from PDF Documents for analysis, archiving, or integration into workflows. However, PDFs can contain diverse content such as running text, tables, and images presented in both machine-readable and scanned formats, making automated data extraction challenging.

# This case study documents the designs and experiments performed to develop an efficient, scalable, end-to-end solution for extracting content from PDFs and converting it into structured markdown format. The study focuses on designing improved OCR/Text-Extraction systems and evaluating various methods to extract text, tables, images, and preserve layouts effectively.

# **TOOL CATEGORIZATION TABLE**

| **TOOL** | **CATEGORY** | **DEFINITION** | **PROS** | **CONS** | **EXTRACTION TYPE** |
| --- | --- | --- | --- | --- | --- |
| **PyPDF2** | Traditional Library | A Python library for splitting, merging, and extracting text from PDF files. | Easy to use, lightweight. | Limited to text extraction, no support for images or complex layouts. | Text extraction |
| **pdfminer** | Traditional Library | A PDF parsing library that converts PDFs into text, while preserving layout and structure. | Detailed extraction, good for well-structured PDFs. | Slower processing, struggles with scanned PDFs. | Text and layout extraction |
| **PyMuPDF (Fitz)** | Traditional Library | A Python binding for MuPDF that handles both text and images, offering fast processing. | Fast and efficient for text and images. | Limited handling of complex layouts or scanned PDFs. | Text and image extraction |
| **pdfplumber** | Traditional Library | A library for extracting tables and text from PDFs while preserving layout. | Excellent for table extraction, preserves layout. | Works best on simpler PDFs. | Text and table extraction |
| **PaddleOCR** | AI Solution (OCR) | An open-source OCR tool based on deep learning that supports multiple languages. | High accuracy, multi-language support. | Slower performance on large documents, requires GPU for best performance. | OCR (text extraction) |
| **pdfrw** | Traditional Library | A library for reading and writing PDF files. It can extract basic text and metadata. | Light and fast for simple tasks. | Limited functionality, no support for tables or images. | Text and metadata extraction |
| **PyPDFium2** | Traditional Library | A Python binding to the PDFium library, used for parsing and rendering PDFs. | Fast, supports rendering and text extraction. | Less support for complex document structures. | Text extraction, rendering |
| **unstructured** | Comprehensive Solution | A library for extracting structured data from unstructured documents, including PDFs. | Highly flexible and adaptable for various types of documents. | Newer tool with less documentation and community support. | Text and structured data extraction |
| **Tabula** | Comprehensive Solution | An open-source tool for extracting tables from PDFs. | Accurate for table extraction, easy to use. | Works best with machine-readable PDFs, struggles with complex layouts. | Table extraction |
| **Camelot** | Comprehensive Solution | A Python library for table extraction from PDFs, focusing on machine-readable and scanned documents. | Handles complex tables, supports multiple extraction strategies. | Struggles with non-table text and complex document layouts. | Table extraction |
| **Marker** | Comprehensive Solution | A tool for detecting and extracting key information, particularly in scanned documents. | Good for structured document analysis and extraction. | Limited to specific extraction types (e.g., forms, signatures). | Key information extraction |
| **EasyOCR** | AI Solution (OCR) | A lightweight OCR tool supporting 80+ languages, good for recognizing text from images and scanned PDFs. | Fast and accurate, supports multiple languages. | Can struggle with noisy backgrounds or low-quality images. | OCR (text extraction) |
| **Unstract** | AI Solution (OCR) | An open-source library for OCR and extracting structured data from PDFs and images. | Focused on OCR and structured data extraction. | Less mature than some other OCR solutions, limited support for complex layouts. | OCR and structured data extraction |
| **OCRmyPDF** | AI Solution (OCR) | A command-line tool that adds OCR text layers to scanned PDF documents. | Easy to use for batch processing, adds OCR metadata. | Limited customization for specific use cases. | OCR (text extraction) |
| **Nougat** | AI Solution (OCR) | A library that provides OCR capabilities combined with document classification features. | Handles both OCR and classification tasks, improving extraction accuracy. | Limited community support, not as widely used as Tesseract or EasyOCR. | OCR and classification extraction |
| **Table Transformer** | Comprehensive Solution | A tool for transforming table data into structured formats such as JSON, CSV, or Excel from PDFs. | Powerful for handling large datasets and complex tables. | Requires careful setup for non-standard table layouts. | Table extraction and transformation |
| **Mineru** | AI Solution (OCR) | An AI-based tool for advanced text recognition and layout detection in PDFs. | Highly accurate with complex document layouts and text. | Not as widely adopted, limited flexibility in document types. | OCR and layout detection |
| **Tesseract** | AI Solution (OCR) | One of the most widely used open-source OCR tools for text extraction from images and scanned documents. | Free, extensive language support, highly customizable. | Accuracy drops with low-quality documents or complex layouts. | OCR (text extraction) |
| **Upstage** | AI Solution (OCR) | A cloud-based OCR service that can process scanned documents, with additional capabilities for handwriting recognition. | Handles complex documents and handwriting recognition well. | Paid service, may be expensive for large datasets. | OCR and handwriting extraction |
| **Amazon Textract** | Cloud Service (OCR) | AWS's OCR service designed to automatically extract text, forms, and tables from scanned documents. | Scalable, integrates well with AWS ecosystem, handles complex layouts. | Costly for high-volume use, requires AWS infrastructure. | OCR, text, tables, forms extraction |
| **Azure Intelligent Search** | Cloud Service (OCR) | A cloud-based service from Microsoft that provides intelligent search and extraction from documents, including PDFs. | High scalability, integration with Microsoft services. | Requires Azure setup and can be expensive for large-scale use. | OCR, text, and metadata extraction |
| **Claude** | AI Solution (OCR) | A model that supports AI-powered extraction and parsing of structured data from PDFs, with strong capabilities in form recognition and data extraction. | Excellent at handling forms and structured data. | Limited support for text-heavy PDFs without specific structures. | OCR, form and structured data extraction |
| **LlamaParser** | AI Solution (OCR) | An OCR-based tool for parsing and extracting text, tables, and key-value pairs from PDFs. | Effective in extracting structured and unstructured data from complex PDFs. | Accuracy can vary with complex, noisy PDFs. | OCR, text, tables, key-value pair extraction |

# **LITERATURE REVIEW AND RESEARCH INSIGHTS**

1. **Mineru: An Open-Source Solution for Precise Document Content Extraction**
   * **Takeaways**:
     + **High-Precision Document Content Extraction:** MinerU leverages advanced PDF-Extract-Kit models, coupled with finely-tuned preprocessing and postprocessing rules, to ensure accurate and consistent content extraction across diverse document types.
     + **Enhanced Performance:** Experimental results highlight that MinerU outperforms existing open-source solutions, delivering significant improvements in the quality and consistency of document content extraction.
2. **A Comparative Study of PDF Parsing Tools Across Diverse Document Categories**
   * **Source**: [arXiv:2409.18839](https://arxiv.org/pdf/2409.18839)
   * **Takeaways**:
     + **Comprehensive Evaluation Across Document Types:**  
       The research compares 10 popular PDF parsing tools across six document categories using the DocLayNet dataset, assessing their text extraction and table detection capabilities. Tools like PyMuPDF and pypdfium2 performed well for text extraction, while Nougat excelled in challenging categories such as Scientific and Patent documents.
     + **Specialized Performance in Table Detection:**  
       Table Transformer (TATR) demonstrated superior table detection performance in categories like Financial, Patent, Law & Regulations, and Scientific documents, whereas Camelot outperformed others in Government Tenders, and PyMuPDF excelled in the Manual category. The study emphasizes selecting tools based on document type and task for optimal results.

# **FILTERED TOOLS FOR IMPLEMENTATION**

**Tools Explored**

The following tools were considered for this study:

**Text Extraction Tools**

1. **PyPDF**: A lightweight library for extracting text from PDFs.
2. **PDFMiner**: A tool for detailed text analysis and extraction.
3. **PyMuPDF (fitz)**: Efficient for extracting text and images from PDFs.
4. **pdfplumber**: Known for handling searchable PDFs with high accuracy.

**OCR Tools (for Scanned PDFs)**

1. **PaddleOCR**: Robust OCR library supporting multi-language extraction.
2. **OCRmyPDF**: Adds OCR layers to scanned PDFs for text extraction.
3. **Tesseract**: Open-source OCR engine (not used here due to access constraints).
4. **EasyOCR**: Lightweight OCR tool supporting multiple languages.

**Table Extraction Tools**

1. **Tabula**: Extracts tables from structured PDFs.
2. **Camelot**: Advanced tool for extracting tables in stream and lattice modes.

**Advanced Tools (Recommended but Not Implemented)**

1. **Nougat**: For extracting structured content like mathematical formulas.
2. **Table Transformer**: Best for highly complex tables.
3. **Claude and LlamaParser**: Large language models (LLMs) for parsing documents.
4. **Amazon Textract and Azure Cognitive Search**: Advanced cloud-based solutions.

# **IMPLEMENTATION**

**Workflow Design**

1. **Preprocessing**
   * Preprocess PDFs (convert to images if scanned).
   * Normalize and enhance quality.
2. **Text Extraction**
   * Utilize **OCRmyPDF** and **PaddleOCR** for scanned PDFs.
   * Use **PyPDF** and **PDFMiner** for machine-readable PDFs.
3. **Table Extraction**
   * Use **PDFPlumber** and **Camelot**.
   * Verify table structure with visualizations.
4. **Image Extraction**
   * Extract images using **PyMuPDF**.

# **CHALLENGES**

# **FUTURE SCOPE**

1. Large Language models and cloud solutions /deep learning models for enhanced layout understanding.
2. To develop a custom preprocessing techniques for noisy scanned PDFs

# **CONCLUSION**

# The combination of traditional tools (e.g., **PyPDF2**, **pdfminer**) and AI-based OCR tools (**Tesseract**, **EasyOCR**) or cloud solutions like amazon textract is essential to tackle the variety of formats present in PDFs